UNIVERSIDAD PERUANA DE CIENCIAS APLICADAS

FACULTAD DE INGENIERÍA

PROGRAMA ACADÉMICO DE INGENIERÍA DE REDES Y COMUNICACIONES

Diseño de una red de área local basada en la arquitectura de redes definidas por software para reducir el tiempo en las configuraciones de un campus universitario

TESIS

Para optar el título profesional de Ingeniero de Redes y Comunicaciones

AUTOR(ES)

Medina Sanchez, Alan Bagner 0009-0008-7611-9186

ASESOR(ES)

Gonzales Figueroa, Renatto Gustavo 0000-0003-3658-3415

Lima, 26 de septiembre de 2023

DEDICATORIA

Este proyecto está dedicado a mi esposa Silenny, mis hijos Kail y Layla y a mis padres Agustín y Gloria, quienes me brindaron todo su apoyo, soporte y comprensión durante todo el desarrollo de este proyecto.

AGRADECIMIENTOS

Agradezco a mi asesor Renatto Gonzales Figueroa por todo el apoyo, sugerencias y recomendaciones para el desarrollo de este proyecto; así como a mi hermosa familia quienes me han apoyado contantemente para continuar y no desistir en el camino.

RESUMEN

En la actualidad, todas las organizaciones son muy dinámicas y se expanden rápidamente, haciendo que constantemente requieran realizar cambios en sus procesos e infraestructura. Parte de los cambios que se realizan en las organizaciones están directamente relacionados a las Redes de Área Local (LAN por sus siglas en inglés), estas redes deben ser escalables, redundantes, altamente disponibles y con la capacidad de ser gestionadas de manera centralizada. Sin embargo, las redes LAN actuales no siempre cumplen todos estos requisitos; por un lado, son muy costosas y difíciles de gestionar de manera centralizada, además estas herramientas dedicadas para esta función solo gestionan a los conmutadores de su propio fabricante, lo cual no permite la posibilidad de tener conmutadores de otros fabricantes operando en la misma LAN debido a que la gestión se tornaría más complejo. Por otro lado, los tiempos para la gestión soy muy elevado, es decir, demandan bastante tiempo para realizar cambios en las configuraciones. En este sentido, este proyecto tiene como propósito hacer una investigación sobre las características y ventajas que brinda SDN para la red LAN de un campus universitario; y en función de ello, realizar un diseño de red que se adecúe a las necesidades de una universidad con un campus en el que involucre una gran cantidad de conmutadores; poder gestionarlos de manera centralizada sin importar el fabricante de estos, además de no representar un costo muy significativo para la consecución de tal fin. Con las pruebas realizadas hemos demostrado que los tiempos para el despliegue de configuraciones y los costos asociados a la gestión centralizada con SDN son muy ventajosos respecto a una red tradicional.

Palabras clave: SDN; gestión centralizada; reducción de costos; controlador ONOS, alta disponibilidad; balanceo de carga

DESIGN OF A LOCAL AREA NETWORK BASED ON THE ARCHITECTURE OF NETWORKS DEFINED BY SOFTWARE TO REDUCE THE TIME IN THE CONFIGURATIONS OF A UNIVERSITY CAMPUS

ABSTRACT

Today, all organizations are very dynamic and expand rapidly, causing them to constantly require changes to their processes and infrastructure. Part of the changes that are made in organizations is directly related to Local Area Networks (LANs), these networks must be scalable, redundant, highly available, and capable of being centrally managed. However, today's LANs do not always meet all of these requirements; On the one hand, they are very expensive and difficult to manage centrally, in addition, these tools dedicated to this function only manage their own manufacturer's switches, which does not allow the possibility of having switches from other manufacturers operating on the same LAN due to because the management would become more complex. On the other hand, management times are very high, that is, they require a lot of time to make changes to the configurations. In this sense, the purpose of this project is to carry out research on the characteristics and advantages that SDN provides for the LAN network of a university campus; and based on this, carry out a network design that adapts to the needs of a university with a campus that involves a large number of switches; to be able to manage them centrally regardless of their manufacturer, in addition to not representing a very important cost to achieve this end. With the tests carried out, we have shown that the configuration usage times and the costs associated with centralized management with SDN are very advantageous compared to a traditional network.

Keywords**:** SDN; centralized management; costs reduction; ONOS controller, high availability; load balancing

u201321407_Alan Bagner Medina Sanchez_Diseño de una red de área local basada en la arquitectura de redes definidas por software para reducir el tiempo en las configuraciones de un campus universitario

INFORME DE ORIGINALIDAD

9%	9%	2%	4%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	repositorioacademico.upc.edu.pe Fuente de Internet	1%
2	dspace.udla.edu.ec Fuente de Internet	1%
3	oa.upm.es Fuente de Internet	1%
4	cybertesis.unmsm.edu.pe Fuente de Internet	1%
5	Submitted to QA Learning Trabajo del estudiante	1%
6	wiki.onosproject.org Fuente de Internet	<1%
7	uvadoc.uva.es Fuente de Internet	<1%
8	repositorio.ug.edu.ec Fuente de Internet	<1%

TABLA DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1. CAPÍTULO 1: ORGANIZACIÓN

1.1. Introducción

Todas las organizaciones actualmente, y desde ya un tiempo atrás, requieren de manera indispensable contar con infraestructura de red que soporte a todos sus servicios informáticos que forman parte de sus procesos. Según Isaac Julio, las redes LAN son quizás lo más importante después del Internet; puesto que se trata de un mecanismo que hace más simple la comunicación entre dos o más computadoras mediante un servidor o computador principal. (Julio, 2015) .

En este sentido, las Universidades, si bien es cierto su objetivo principal es el de impartir educación superior universitaria, para poder lograr este objetivo requiere de servicios informáticos que sean estables, eficientes, seguros y escalables. Y para ello, es necesario que puedan contar con redes LAN que soporten los cambios que contantemente tienen que hacer en sus infraestructuras físicas y procesos para soportar al crecimiento por la demanda de estudiantes; así, de esta manera poder incrementar el nivel de satisfacción de sus usuarios administrativos, estudiantes y docentes.

Debido al contante crecimiento de las redes LAN, cada vez es más difícil llevar a cabo una adecuada gestión, pues deben tener la capacidad de dar acceso a los servicios informáticos de forma eficiente. Debido a que estos servicios crecen en número y complejidad (PowerData, 2016), las redes LAN también han tenido que seguir este ritmo y las organizaciones se han visto obligadas a tener que mejorarlas; y como consecuencia, se han generado diversas dificultades a nivel de la gestión, seguridad, integración y optimización de estas.

Según Ante ello, como alternativa para solucionar estas dificultades, existe la arquitectura de Redes Definidas por Software (SDN - Software Defined Network), la cual tiene como objetivo tener una arquitectura dinámica, rentable y adaptable; por lo tanto, una herramienta ideal para administrar grandes redes de datos e implementar aplicaciones personalizadas para diferentes tipos de requisitos de redes de comunicación (Sicrom, 2018).

Este proyecto, basado en las características que la arquitectura de red SDN ofrece para la gestión de grandes redes de datos, tiene como finalidad aplicarlo en el diseño de una red para un Campus Universitario.

1.2. Organización objetivo

La organización objetivo para el desarrollo del presente proyecto es un campus universitario. Las universidades, públicas y privadas tienen como propósito el brindar servicio de educación superior de nivel universitario y post grado; para poder conseguir este propósito requieren de infraestructura física para ambientes académicos y administrativos (aulas, laboratorios, oficinas, bibliotecas, etc.), infraestructura tecnológica (computadoras, proyectores, servidores, conmutadores, etc.), servicios informáticos (Intranet, Biblioteca Virtual, Aula Virtual, Servicio de Internet, Servicio WiFi, etc.), mobiliario (escritorios, sillas, carpetas, etc.) y personal docente y administrativo.

Según la SUNEDU, en su informe Bienal sobre la realidad Universitaria en el Perú, las universidades, debido a la demanda por la educación superior universitaria que existe, en las dos últimas décadas se han caracterizado por tener un constante crecimiento en infraestructura (SUNEDU, 2018)

El dictado de clases, generalmente se desarrolla en ambiente académicos (aulas y/o laboratorios) los cuales están implementados con proyectores, pizarras, computadoras, etc. los mismos que tienen que estar interconectados mediante la red LAN. Sin embargo, en muchas universidades para complementar el dictado de clases en las Aulas dentro de sus campus universitarios, también cuentan con Aulas virtuales a través de internet. Por lo tanto, para poder brindar un servicio académico de calidad y soportar todos sus servicios informáticos, las universidades requieren de una infraestructura de red que responda las exigencias y necesidades.

De acuerdo con lo reportado por la SUNEDU, Superintendencia Nacional de Educación Superior, existen 143 universidades en el Perú entre privadas y públicas; muchas de las cuales, cuentan con más de un campus universitario, es decir; son universidades multicampus con presencia tanto a nivel de Lima y Provincias.

1.2.1. Campo de acción

Este proyecto está orientado a un entorno universitario, el mismo que comprende la red LAN que brinda conectividad al personal administrativo, personal académico (estudiantes y profesores) y periféricos de un campus universitario. Las redes LAN de este tipo de

organizaciones actualmente tienen un diseño bajo una arquitectura del tipo jerárquica de tres niveles: conmutadores de core, distribución y acceso.

Las universidades; para poder soportar todos sus servicios informáticos, cuentan con redes LAN en cada uno de sus campus. Estas redes LAN se encuentran interconectadas mediante redes WAN con la finalidad de tener servicios informáticos centralizados y/o sincronizados y de esta manera, poder tener la capacidad de brindar el mismo tipo y nivel de servicio en todos los campus sin que sus usuarios sientan diferencias entre uno u otro campus universitario.

Los conmutadores que forman parte de sus redes LAN, en el nivel de core, junto a otros equipos de redes (controladores wifi, servidores de autenticación, servidores de monitoreo, etc.) y servicios de TI principales de la universidad se encuentran ubicados en un Centro de Datos; mientras que los conmutadores de distribución y acceso se encuentran ubicados en distintos pabellones y pisos dentro del mismo campus universitario.

Por lo tanto, el proyecto tendrá como alcance la modificación del diseño jerárquico de la red de datos local mediante el uso de estándares y buenas prácticas que la tecnología elegida ofrece.

1.3. Identificación del problema

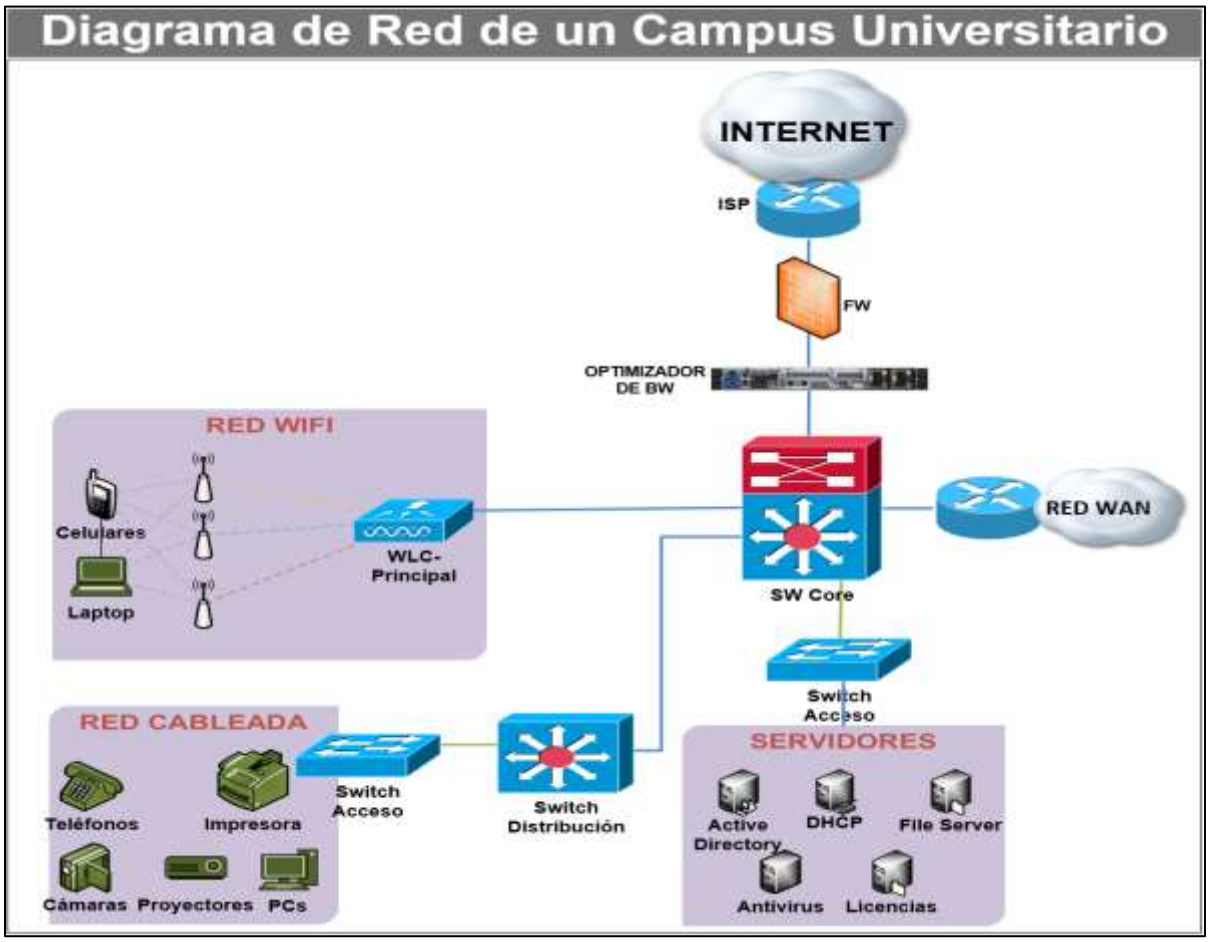
1.3.1. Situación problemática

Las universidades tienen un constante crecimiento en cantidad de personal académico y administrativo, así como en infraestructura. Esto las obliga a que junto a este crecimiento también hagan actualizaciones en sus redes de datos de manera constante.

Basado en mi experiencia como profesional y estudiante, las universidades en cada uno de sus campus, generalmente, tienen una infraestructura de red idéntica, es decir, están diseñadas bajo una arquitectura de red tradicional jerárquica.

Figura 1

Diagrama general de la red de un Campus Universitario



Tal como se muestra en el diagrama de la *figura 1*, cada campus cuenta con su propia salida a internet y un enlace redundante para la interconexión entre campus universitario identificado como ***RED WAN*** en el diagrama. En este diagrama se puede identificar la existencia de los tres niveles jerárquicos de la arquitectura: Conmutadores de acceso en el nivel más bajo, los cuales concentran a los dispositivos finales como impresoras, teléfonos, cámaras IP, Computadoras, etc.; Conmutadores de distribución en nivel intermedio, estos concentran a los conmutadores del nivel de acceso, y finalmente en nivel superior están los conmutadores de Core que concentran a los conmutadores de distribución, los controladores WiFi y los ruteadores del ISP.

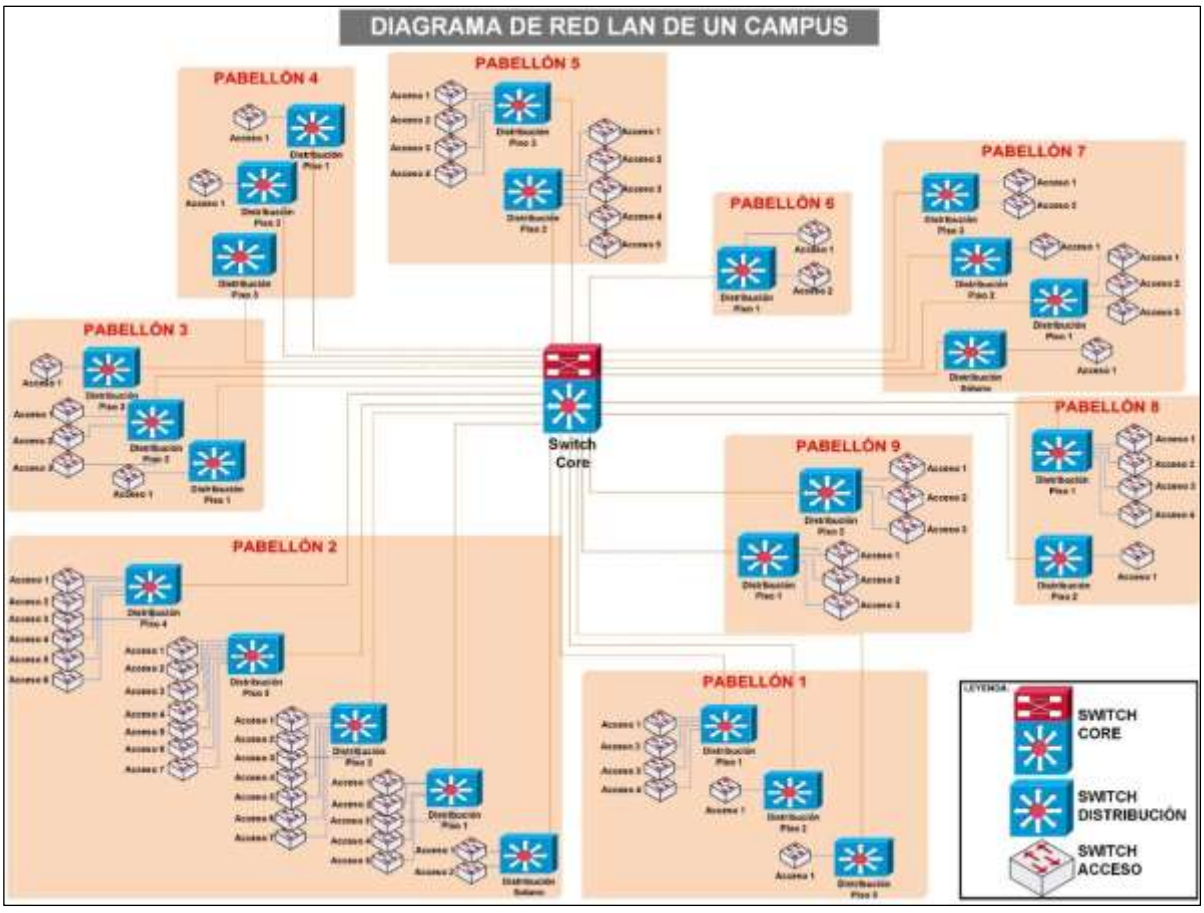
A pesar de que este tipo de universidades cuentan con una Red WAN para interconectar sus campus y tener comunicación directa entre ellos, no siempre cuentan con la gestión centralizada de los equipos de red LAN.

Entre las principales características que se encuentran en este tipo de redes LAN están las siguientes:

- ✓ **Gran Cantidad de conmutadores:** La cantidad de conmutadores de capa 2, según el modelo OSI, que forman parte de este tipo de redes en cada uno de los campus universitarios es mayor a 100, los cuales brindan servicio de conectividad de red en los diferentes pabellones del campus a antenas WiFi, computadoras de aulas y laboratorios, proyectores y múltiples dispositivos finales.
- ✓ **Segmentación**: En cada uno de los campus, la red LAN está segmentada por pabellones, tipo de usuarios (administrativos, docentes o estudiantes) o inclusive por la forma en que se brinda el servicio (cableada o inalámbrica); esta segmentación está dada por redes virtuales (VLAN) y se tienen más de 70 VLAN por campus universitario. Así mismo, en los conmutadores del nivel de Core, se encuentran aplicadas políticas de acceso y restricción a través de Listas de Control de Acceso.

Figura 2

Diagrama de Red LAN referencial de un Campus Universitario



Basado en la gran cantidad de equipos, VLANs, políticas y el tamaño de las redes, la gestión se torna muy compleja lo cual puede llevar a incurrir en posibles situaciones problemáticas:

a) Tiempos muy elevados para el despliegue de configuraciones

Los tiempos utilizados, por el personal que gestiona el servicio, para realizar un cambio en la arquitectura de red son bastante elevados; esto implica que el administrador dedique gran parte de su tiempo a la gestión y configuración de equipamiento. Esto sin duda deja muy poco tiempo para la búsqueda de alternativas de mejora y nuevas soluciones para la red.

A continuación, se detalla una tabla con ejemplos de: tipo de configuraciones y los tiempos que toma realizarlas.

Tabla 1

Medición de Tiempo en la gestión de equipamiento en una red tradicional

Actividad	Detalle	Procedimientos	Tiempo (minutos)	Riesgos	Tiempo Total (minutos)
Crear una nueva VLAN	Crear una nueva VLAN y realizar el despliegue la misma en la red.	Conectarse al conmutador Core y crear la VLAN	**30**	Si no se identifica los conmutador y puertos y luego no se aplica la configuración correctamente, podría quedar sin servicio de red un área, un pabellón o inclusive todo el Campus Universitario.	**120**
		Identificar el(los) conmutador(es) y el(los) puerto(s) para crear y configurar VLAN	**60**		
		Conectarse al(los) conmutador(es) para configurarlos			
		Configurar la VLAN en el(los) puerto(s) del conmutador de manera correcta	**30**		

Actividad	Detalle	Procedimientos	Tiempo (minutos)	Riesgos	Tiempo Total (minutos)
Crear un nuevo segmento de red	Crear un nuevo segmento de red para una nueva área, servicio o proyecto	Identificar el nuevo segmento de red a ser creado	**45**	Si no se aplica la configuración de la VLAN de manera correcta, podría dejar sin servicio de reda un área, un pabellón o inclusive a todo el campus universitario.	**150**
		Conectarse al conmutador Core y crear el nuevo segmento de red	**45**		
		Identificar la interfaz física o VLAN a la cual se aplicará el nuevo segmento de red.	**30**		
		Crear la VLAN en caso aún no exista	**30**		
Cambiar de VLAN a puertos de un conmutador	Actualizar la VLAN en uno o múltiples puertos de un conmutador	Identificar el(los) conmutador(es) y el(los) puerto(s) para crear y configurar VLAN	**60**	Si no se aplica la configuración de la VLAN de manera correcta, podría dejar sin servicio de reda un área, un pabellón o inclusive a todo el campus universitario.	**90**
		Conectarse al(los) conmutador(es) para configurarlos			
		Configurar la VLAN en el(los) puerto(s) del conmutador de manera correcta	**30**		
Implementar un nuevo conmutador	Implementación de un nuevo conmutador o reemplazo por uno que falló	Configuración básica (IP, Comunidad SNMP, NTP, ETC) del conmutador	**45**	Si no se aplica la configuración al conmutador de manera correcta, podría no tener servicio el área involucrada, un pabellón o inclusive dejar sin servicio de red a todo el campus universitario.	**225**
		Instalación (rackeo) del conmutador en su ubicación física	**60**		
		Configuración específica del conmutador	**120**		

Los tiempos que se muestran en la Tabla 1, están ajustados a la realidad de un campus universitario con una red compleja del nivel jerárquico (conmutadores de Core, Distribución y Acceso) con un administrador de red dedicado a estas funciones y considerando equipamiento de backup en espera.

b) Necesidad de personal altamente capacitado

El despliegue de nuevas configuraciones y/o actualizaciones se torna complicado debido a que se tiene que ingresar a cada uno de los equipos involucrados de manera individual, y en cada uno ejecutar los comandos requeridos. Para realizar estas configuraciones se requiere de personal con conocimientos avanzados y bastante experiencia en la infraestructura; de lo contrario, se generarán problemas de funcionamiento en toda la red.

Así mismo, para realizar nuevas configuraciones o cambios que impliquen cierta complejidad es necesario se cuente con personal de manera presencial en el mismo lugar donde están instalados los equipos. Esto, con la finalidad de actuar de manera rápida y oportuna ante una posible incidencia y evitar pérdida del servicio de red.

c) Software de gestión muy costoso o ausencia de este

Para la gestión de manera centralizada de una red LAN existen soluciones propietarias. Por ejemplo, para una universidad con equipamiento de la marca Cisco, el Fabricante cuenta con su propio software de gestión centralizada. Sin embargo; primero, el costo por la compra e implementación del software, soporte y licenciamiento son bastante elevados.

A continuación, se detalla un cuadro con los costos de licenciamiento y soporte anual; inclusive, sin considerar el costo de implementación, este ya representa una inversión bastante elevada.

Figura 3

Costo por licenciamiento y soporte para el Cisco Prime Infrastructure

Software de Gestión	Tipo de licenciamiento	N° de Licencias requeridas	Costo por licenciamiento	Precio por el Soporte Anual	Precio Total
Cisco Prime Infrastructure	Este software se liencia por equipo o dispositivo gestionado	120	$12,600.00	$12,409.72	$25,009.72

Nota. Los precios a los que se hacen referencia en la Figura 3 no incluyen el I.G.V.

Y segundo, estos softwares no siempre se terminan alineando a las necesidades de la organización debido que no son softwares diseñados en base a las necesidades específicas y la consecuencia de ello es que terminan siendo utilizados únicamente como un software de monitoreo; una de sus funciones básicas de estas soluciones.

Otra consideración muy importante a tener en cuenta en la búsqueda de una gestión centralizada con este tipo de soluciones es que la universidad debe tener todo su equipamiento bajo un único fabricante; sólo así se podrá garantizar la compatibilidad entre los dispositivos gestionados y el software que los gestiona. De esta manera la universidad es dependiente del fabricante de equipos y software de gestión.

1.3.2. Problema a resolver

El principal problema a resolver es la falta de **gestión centralizada** y **los altos tiempos que demanda el despliegue de configuraciones y actualizaciones** en los conmutadores que conforman la red LAN los cuales brindan conectividad a los usuarios y dispositivos finales.

1.4. Objetivo General y Objetivos Específicos

1.4.1. Objetivo General

Diseñar una red LAN con un sistema de configuraciones automatizadas, basada en SDN y usando las buenas prácticas de la Open Network Fundation (ONF), con el propósito de reducir los tiempos despliegue de configuraciones y los costos en la gestión centralizada para un Campus Universitario.

1.4.2. Objetivos Específicos

1. Gestionar todos los conmutadores que conforman la red desde un punto central, con la finalidad de que todos sean monitoreados y gestionados de manera centralizada.
2. Mantener un arreglo de controladores SDN en alta disponibilidad y permita el balanceo de carga en la gestión de los conmutadores, de tal manera que estos no pierdan conectividad con el controlador.
3. Disminuir los tiempos de despliegue de configuraciones y actualizaciones de la red.
4. Proponer una optimización en los costos de inversión para la implementación de red LAN con su gestión centralizada.

1.4.3. Indicadores de logro de los objetivos

Tabla 2

Indicadores de logro de Objetivos

ID	Objetivo Específico	Indicador de Logro	Métrica
OE1	1. Gestionar todos los conmutadores que conforman la red desde un punto central, con la finalidad de que todos sean monitoreados y gestionados de manera centralizada.	1.1. Inventario del 100% de dispositivos (conmutadores y host) que integran la red LAN.	✓ Número o porcentaje de dispositivos inventariados desde el controlador
		1.2. Monitoreo de todos los dispositivos (conmutadores y host) que conforman la red.	✓ Número o porcentaje de dispositivos monitoreados por el controlador SDN
OE2	2. Mantener un arreglo de controladores SDN en alta disponibilidad y permita el balanceo de carga en la gestión de los conmutadores, de tal manera que estos no pierdan conectividad con el controlador.	2.1. Dos o más nodos de controlador SDN implementados para brindar la alta disponibilidad y balanceo de carga en la gestión de los conmutadores.	✓ Número de Nodos de controlador implementados para brindar la alta disponibilidad y balanceo.
		2.2 Correcto funcionamiento de la alta disponibilidad y balanceo cuando un nodo del controlador SDN falla.	✓ Pruebas de falla en por lo menos un nodo del controlador SDN
OE3	3. Disminuir los tiempos de despliegue de configuraciones y actualizaciones de la red.	3.1. Reducir hasta en un 40% el tiempo ocupado en una Actualización de la configuración de la red con SDN respecto a la red tradicional.	✓ Tiempo o porcentaje que toma la actualización de la configuración de la red con SDN

ID	Objetivo Específico	Indicador de Logro	Métrica
OE4	4. Proponer una optimización en los costos de inversión para la implementación de red LAN con su gestión centralizada.	4.1. Reducir hasta en un 30% el costo asociado a la gestión centralizada de la red SDN, respecto de una red tradicional implementada con equipamiento Cisco y Fortinet.	✓ Porcentaje de costos asociados a la gestión centralizada de la red SDN

1.5. Justificación

Interoperabilidad de equipos entre diferentes fabricantes: Las universidades dejarán de ser dependientes de un único fabricante de equipamiento (conmutadores) para mantener la compatibilidad con sus softwares de gestión. El proyecto busca establecer la integración de áreas específicas de redes y comunicaciones como el despliegue de configuraciones y la gestión de equipamiento de manera centralizada haciendo uso de una arquitectura y protocolo establecido para esta tendencia tecnológica.

Reducción de costos en adquisición de equipamiento: Las universidades podrán implementar nuevos campus universitarios y/o pabellones o simplemente hacer una remodelación sin la necesidad de pensar en un nuevo software de gestión y/o la cantidad de licencias necesarias para soportar al nuevo equipamiento. Inclusive puede optar por comprar equipos de un fabricante diferente que le ofrezca equipos con nuevas o mejores características a los que ya tiene.

Reducción de horas hombre dedicadas a la gestión de la red LAN: El optar por una nueva arquitectura de red LAN basada en SDN, puede hacer que las universidades puedan evitar la contratación de nuevo personal para la gestión de la red, o inclusive optimizar el tiempo del personal existente. Al tener una red centralizada hace posible el poder gestionar más equipos de red con menos personas.

1.6. Estado del arte

1.6.1. Antecedentes

Las computadoras personales comenzaron a aparecer en el mercado a fines de la década de 1970; estas estaban destinadas para uso personal, hasta ese momento sin la necesidad ni tener

en cuenta en la conexión a una red informática. Aproximadamente 10 años más tarde, a principios de la década de 1980, surgieron varias arquitecturas de red con la finalidad de satisfacer las necesidades de comunicación en las organizaciones, es decir en un ambiente de red de área local (LAN) (Martínez et al., 2016).

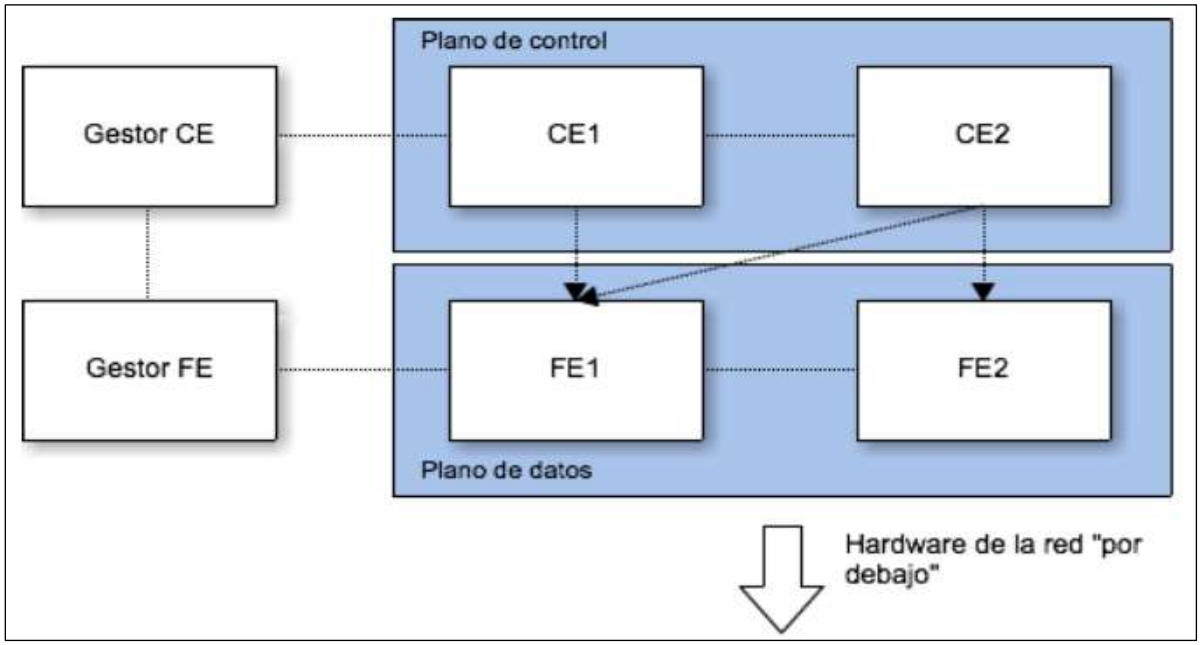
En los últimos años, el término Software Defined Networking – SDN, ha tomado relevancia en el mundo tecnológico y directamente en las redes LAN y WAN. No obstante, este concepto surgió en el año 1996, a raíz de la necesidad de contar con un controlador que se encargue del control integral del reenvío de paquetes en los nodos de la red de los ISP (proveedor de servicios de internet).

A continuación, se describen algunos trabajos de quienes iniciaron con las investigaciones, aunque no están limitadas necesariamente a las siguientes propuestas, contribuyen al desarrollo de esta nueva arquitectura de red:

- Los miembros de la IETF (Internet Engineering Task Force), presentan la RFC1987 en la cual se proponen el protocolo GSMP (General Switch Management Protocol) en su versión 1.1; esta versión permite tener el control de manera centralizada de los conmutadores en las redes ATM, esto facilita el mantenimiento y operación de la red. (Newman et al., 1996)
- El grupo de trabajo de IETF (Internet Engineering Task Force), presentan la RFC2297 en la cual se propone el protocolo GSMP (General Switch Management Protocol) versión 2.0; esta versión permite tener el control de manera centralizada de los conmutadores en las redes ATM, esto facilita el mantenimiento y operación de la red. (Newman et al., 1998)
- El grupo de trabajo de la IETF, presentan la RFC3746 en la cual se separa los planos de control y los planos de datos en los nodos de la red, es decir, se separa el elemento de control del reenvió de paquetes, pero siempre interconectados interconectándolos a través de una interfaz estándar de comunicación de nombre ForCES (Yang et al., 2004).

Figura 4

Arquitectura ejemplo en ForCES



Nota. De "Redes Definidas por Software (SDN): OpenFlow", por Serrano Carrera, 2015 (https://riunet.upv.es/bitstream/handle/10251/62801/SERRANO%20-%20Redes%20Definidas%20por%20Software%20(SDN):%20OpenFlow.pdf?sequence=3).

- En el documento presentado por Luo Lifeng y Eric Wood, proponen una arquitectura nueva de gestión de seguridad de conmutadores, esta propuesta se basa en el flujo mediante un controlador centralizado que gestiona las rutas de los flujos, al cual se le llama Ethane (Luo & Wood, 2007).

En el año 2011, la ONF (Open Networking Foundation por sus siglas en inglés) logra estandarizar el protocolo Openflow, el mismo que está definido en los documentos "OpenFlow Conmutador Specification", y el 2018 fue lanzada la versión 1.1.0 del protocolo, Sin embargo, este protocolo tuvo su inicio de desarrollo en el 2008 en la Universidad de Stanford. Openflow es el protocolo más utilizado y aplicado en soluciones de redes definidas por software, ya que permite que las entradas se configuren en flujos de tabla y se envíen a un controlador central (Chafloque Mejía, 2018).

En el año 2011, grandes corporaciones informáticas como Facebook, Deutsche Telekom, Microsoft, Google, Yahoo! y Verizon crearon la *Open Network Foundation* (ONF), una organización creada sin fines de lucro con el principal objetivo de promover la adopción e innovación en redes definidas por software y promoviendo el desarrollo de estándares de

código libre y apoyándose en el trabajo previo que fue desarrollado por investigadores de universidades con Berkeley y Stanford. Luego esta organización es la que propone el protocolo OpenFlow el que se detallará más adelante (Andrés et al., 2015).

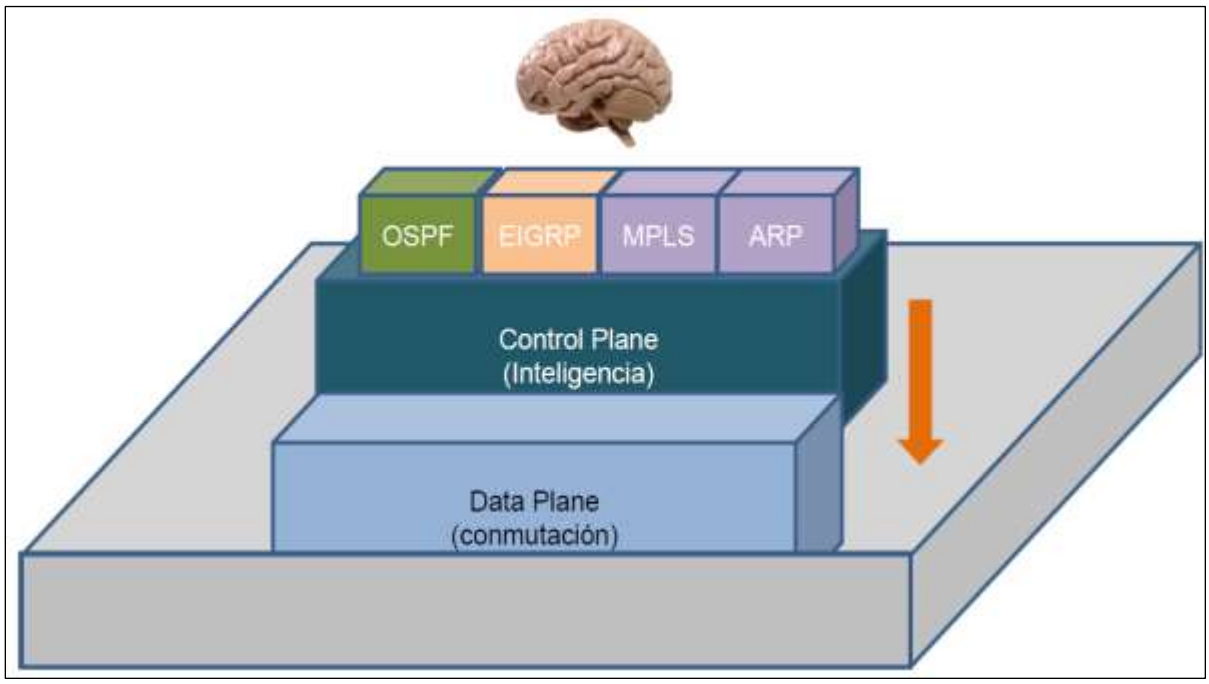
La finalidad principal de las redes SDN es muy clara: en los equipos de red físicos (que pueden ser conmutadores o enrutadores), es la de separar el plano de control del plano de datos y combinarlos en un solo elemento (controlador) fuera de la red física (Andrés et al., 2015).

En relación a lo descrito, a continuación se mostrará de manera gráfica la composición de una red clásica (tradicional) y una red basada en software (SDN), en donde se puede visualizar claramente las diferencias de cómo operan la inteligencia (plano de control). Mientras que en una red clásica (tradicional) el plano de control está dentro de cada uno de los equipos (conmutadores), en una red basada en software (SDN) está dentro de un equipo central llamado controlador.

Composición de una red Tradicional

Figura 5

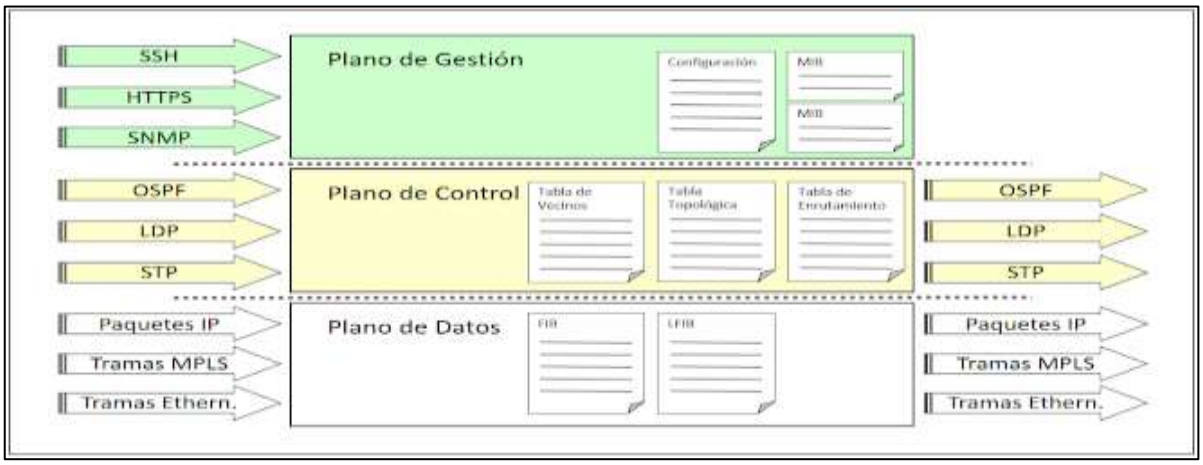
Composición de un dispositivo de red tradicional



Nota. De "Redes definidas por Software (SDN)", por Ccoyllo Sulca, 2018 (https://informatica.ucm.es/data/cont/media/www/pag-103596/transparencias/redes-por-software-SDN.pdf).

Figura 6

Planos de Operación de un dispositivo de Red



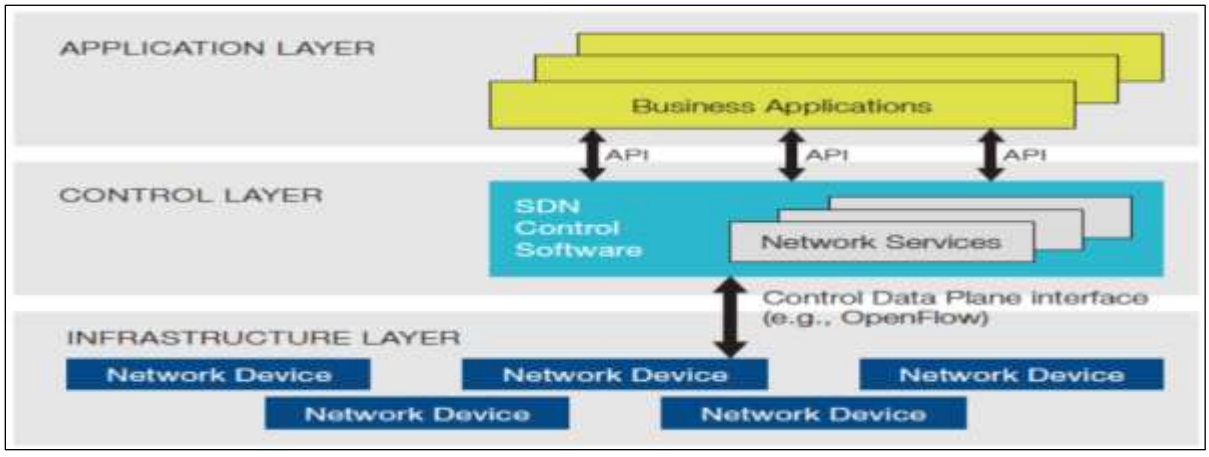
Nota. De "Los planos de operación de un dispositivo de red", por Gerometta, 2013 (http://librosnetworking.blogspot.com/2013/06/los-planos-de-operacion-de-un.html).

Composición de una red SDN

En una red SDN, su arquitectura consta de tres capas: capa de infraestructura, capa de control y capa de aplicación tal como se puede apreciar en la Figura 7, estas capas interactúan a través de interfaces abiertas, hacia el sur (Southbound) y hacia el norte (Northbound) (España Tarapuez, 2016).

Figura 7

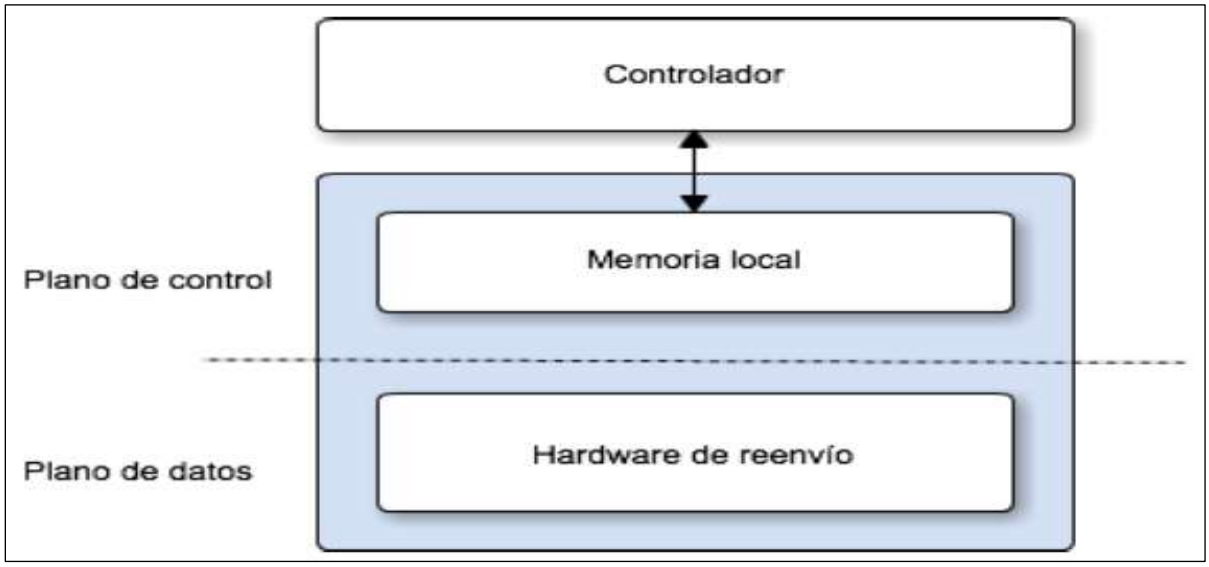
Arquitectura SDN



Nota. De "Diseño y simulación de una red definida por software (SDN)", por España Tarapuez, 2016 (https://www.academia.edu/82308978/Dise%C3%B1o_y_simulaci%C3%B3n_de_una_red_definida_por_software_SDN_).

Figura 8

Dispositivo de red en la capa de infraestructura



Nota. De "Redes Definidas por Software (SDN): OpenFlow", por Serrano Carrera, 2015 (https://riunet.upv.es/bitstream/handle/10251/62801/SERRANO%20-%20Redes%20Definidas%20por%20Software%20(SDN):%20OpenFlow.pdf?sequence=3).

1.6.2. Actualidad

En la actualidad, por un lado, tenemos a las redes de datos tradicionales; redes con una gran cantidad de conmutadores, los cuales en algunos casos no son gestionados y en otros, tienen que ser gestionados con herramientas de gestión propietarias de diversos fabricantes las mismas que gestionan únicamente a conmutadores de su misma marca. Y, por otro lado, tenemos a las redes SDN que ya tienen de manera predeterminada el concepto de gestión centralizada.

Cómo bien lo comentamos en los antecedentes, el concepto de redes SDN no es reciente, sin embargo, actualmente se considera un concepto completamente disruptivo en las Redes de Área Local.

De acuerdo con el artículo Comparativa entre red tradicional y red definida por software: Caso de estudio ESPAM MFL de la Revista Ibérica de Sistemas y Tecnologías de la Información (RISTI) del 2020, este estudio demuestra que las redes SDN brindan un mejor rendimiento que las redes tradicionales al disminuir la cantidad de dispositivos indispensables para operar la red. Asimismo, la gestión de las redes SDN se vuelve más sencilla gracias a la intervención de un controlador que ejecuta y centraliza toda la inteligencia de la red (Alcívar & Navia, 2020).

Las redes SDN pueden brindar un mejor rendimiento que las redes tradicionales, ofrecen una menor latencia de acuerdo con el estudio y pruebas realizadas por (Alcívar & Navia, 2020), el resultado de las pruebas arrojó que los tiempos de la latencia en SDN son más bajos que los de la red tradicional (picos de 4.15ms versus 8.95ms respectivamente). Así mismo, por un lado, se resalta la relevancia e importancia que esta tecnología tiene en la actualidad; en Ecuador varias universidades cuentan con investigaciones y desarrollos de prototipos para pasar de una infraestructura de red clásica (tradicional) a SDN y, por otro lado, también se realza la importancia de las redes SDN por su rentabilidad frente a las redes tradicionales puesto que no se depende más de una solución propietaria.

Chafloque Mejía (2018), en su Tesis para optar por el Título Profesional de Ingeniero de Telecomunicaciones, basa su investigación en el controlador Opendaylight, haciendo uso del RESTCONF como protocolo. Realizó su trabajo basado en software de código abierto, mediante lo cual realiza una automatización de las redes y le agrega seguridad (Chafloque Mejía, 2018).

Según los autores del artículo de investigación en la revista Pro Sciences, una de las características que encontraron como parte de sus pruebas durante una simulación de una red SDN en Mininet, mencionan que estas redes permiten el reúso de los equipos lo cual se traduce en una reducción de los costos y presupuestos de inversión y el menor uso de energía eléctrica para los dispositivos. Como parte de su investigación pudieron demostrar en la parte práctica que, al tener un solo equipo físico en el que se crean de manera virtual múltiples enrutadores (se utilizó MiKroTik) para gestionar la red, esto genera una reducción de costos en la operación y ejecución de la red (Vega Guallpa et al., 2022).

Una de las principales funciones que brindan las redes SDN es la virtualización de dispositivos de red, creando así una red inteligente capaz de emular múltiples redes interconectadas; esto lo pudieron confirmar emulando redes LAN y WAN mediante la configuración de un enrutador MiKroTik virtualizado, este brindó acceso a internet a los dispositivos que estaban en la red; de esta manera se evidenció los beneficios de optimizar la eficiencia, el rendimiento, la escalabilidad, la flexibilidad y la programabilidad de los dispositivos (Vega Guallpa et al., 2022).

1.6.3. Tendencias

Sin duda alguna, las organizaciones cada vez son más dependientes de la tecnología; con lo cual las redes modernas ya no pueden tan siquiera por un momento tener tiempos de inactividad. Pues una falla en un sistema informático puede provocar un desastre permanente o por lo menos afectar la satisfacción de sus clientes o impactar en su reputación ante ellos. Así mismo, tomarse demasiado tiempo para implementar cambios en la infraestructura de la red puede obstaculizar el progreso de una empresa. Ante la necesidad de tener el servicio de red de datos disponible de manera permanente, también surge la necesidad de tener un sistema que pudiese realizar actualizaciones inteligentes en tiempo real.

SDN es una arquitectura de red con una tecnología que permite gestionar la red desde una capa por encima de la capa física que actúa como un controlador de red y es independiente de los fabricantes, es decir, que se pueden integrar dispositivos de red de diferentes fabricantes). Esta tecnología SDN también puede integrarse con otro tipo de soluciones, como aplicaciones que analizan el tráfico generado en la infraestructura de red y sacarle provecho inclusive mediante la generación de publicidad dirigida hacia los clientes de la red dentro de la organización (Ávila Martín, 2017).

Las redes SDN, cada vez más están siendo requeridas e implementadas en las organizaciones debido a sus bondades que ofrece, como, por ejemplo:

- **Flexibilidad y agilidad**: Uno de los principales beneficios que obtenemos de SDN es que no necesitamos ir a cada uno de los dispositivos (conmutadores o enrutadores) para configurarlo de manera independiente, sino por lo contrario, únicamente se podrá hacer desde la consola de un punto centralizado, el controlador, y este se encargará de desplegar la configuración a todos los dispositivos que componen la red (Ávila Martín, 2017). Las reglas de enrutamiento, políticas nuevas o existentes, fácilmente se pueden implementar y actualizar desde el controlador.

- **Automatización y centralización**: En SDN al contar con un controlador, facilita la administración centralizada de las configuraciones, el monitoreo y el control de la red con lo cual este trabajo se hace más eficiente. Así mismo, SDN participa como un coordinador de tareas al cambiar o agregar nuevas redes y cualquier objeto de red. Es así como se consigue optimizar los tiempos dedicados a la gestión de la red, pues los administradores de

la red tendrán más tiempo para dedicarlo a funciones diferentes de configuración de la red (Ávila Martín, 2017).

- **Seguridad**: La seguridad es una parte indispensable que se debe tener en cuenta al separar los planos de control y datos; esto genera nuevos desafíos que no se pueden dejar pasar por alto. En este punto es donde se incorpora la funcionalidad de la microsegmentación de la red, esta es una estrategia para dividir la red en segmentos más pequeños e independientes (aislados) y limitar la comunicación entre los diferentes segmentos, esta segmentación de la red se logra a través del control y programación centralizada desde el mismo controlador mediante la generación de políticas a nivel de paquetes o flujos; su funcionamiento es diferente a la segmentación por VLAN. Esta es una manera que ha servido para fortalecer la seguridad en las redes SDN, pues de no hacerlo se estaría dando lugar a que los hackers utilicen algún dispositivo de la red que sea blanco fácil y desde ahí atacar a los principales (Ávila Martín, 2017).

Así mismo, a medida que pasa el tiempo las exigencias hacia las redes tradicionales y SDN son cada vez más grandes y complejas pues las organizaciones manejan cada vez volúmenes más grandes de información. Actualmente existen aplicaciones que gestionan grandes volúmenes de datos como por ejemplo "Big Data" así como una inmensa variedad de información que hacen que las redes tradicionales sean vean desbordadas para gestionarlo; por ejemplo, transmisión de video en alta definición, la realidad virtual, streaming de música y el internet de las cosas que son bastante demandantes de ancho de banda y dinámicos y por lo tanto requieren de redes de alta velocidad y adaptables (Andrés et al., 2015).

Para tener como referencia las exigencias en cuanto las velocidades que deben estar soportar las redes y por tanto deben estar preparadas para ello, revisaremos el crecimiento del tráfico de datos en el mundo en los últimos años y cuál es la tendencia del mismo hacia los siguientes años. De acuerdo con el informe de internet anual de Cisco 2018-2023, el incremento de las aplicaciones y redes móviles proyectadas hasta el 2023 son datos a tener en cuenta:

- Más del 70% de la población mundial (aproximadamente 5.7 billones de personas) tendrá conectividad a través de un dispositivo móvil (2G, 3G, 4G o 5G).
- El 66% de la población mundial (aproximadamente 5.3 billones de personas) serán usuarios de internet; de estos, se estima que, por persona, 3.6 dispositivos estarán

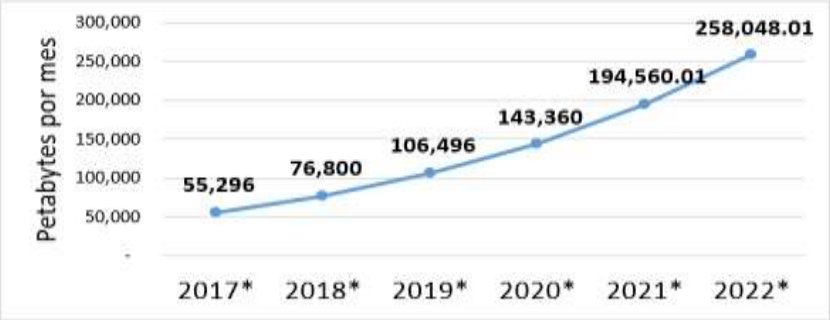
conectados a la red; del total de dispositivos (aproximadamente 19.08 billones), el 47%, es decir casi la mitad tendrán capacidades de transmisión de video.

En la siguiente tabla, se muestran datos históricos que hacen referencia a la evolución en el incremento del tráfico de datos (video), como se puede observar la tendencia es en aumento con porcentajes bastante elevados. Con lo cual podemos afirmar que las exigencias para las redes actuales son bastante complejas y estas tienen que responder a tales exigencias, pero también son oportunidades y desafíos que tienen que saber afrontar, en este caso preciso las redes SDN.

Figura 9

Tráfico mundial en internet en entrega de contenidos 2017-2022

Año	Petabytes por mes
2017*	55,296
2018*	76,800
2019*	106,496
2020*	143,360
2021*	194,560.01
2022*	258,048.01



Nota. Adaptado de "Previsión: tráfico mundial en Internet por red de entrega de contenidos 2017-2022", por Fernández, 2020 (https://es.statista.com/estadisticas/635666/prevision-trafico-mundial-en-internet-por-red-de-entrega-de-contenidos/).

El tráfico que se muestra en la figura 9, incluye la entrega de videos (Fernández, 2020).
En la siguiente imagen se puede apreciar la tendencia en crecimiento en cuanto al tráfico por tipo de red, si bien es cierto que la gráfica muestra datos sólo hasta el 2019, en base a la tendencia mostrada, hace inferir que en los últimos años y hacia el futuro el crecimiento de tráfico irá en aumento, esto debido a la infinidad de aplicaciones y nuevos desarrollos que se tienen constantemente.

Figura 10.

Tráfico de Internet en base al tipo de red

Nota. De "Redes Definidas por Software (SDN): OpenFlow", por Serrano Carrera, 2015 (https://riunet.upv.es/bitstream/handle/10251/62801/SERRANO%20-%20Redes%20Definidas%20por%20Software%20(SDN):%20OpenFlow.pdf?sequence=3).

Así mismo, también podemos observar en la siguiente imagen la demanda del tráfico por tipo de servicio o segmento; en este caso se puede apreciar que el tráfico que más consume ancho de banda es el tráfico de video por internet. Tal como se pudo ver en la figura 10, este tipo de tráfico también tiene una tendencia al crecimiento de una manera muy significativa.

Figura 11

Tráfico de Internet en base al tipo de contenido

Nota. De "Redes Definidas por Software (SDN): OpenFlow", por Serrano Carrera, 2015 (https://riunet.upv.es/bitstream/handle/10251/62801/SERRANO%20-%20Redes%20Definidas%20por%20Software%20(SDN):%20OpenFlow.pdf?sequence=3).

Por lo tanto, en base a los datos mostrados en la ***Figura 9*** y en las ***Figuras 10*** y ***11***, es importante que las organizaciones se preocupen por tener o implementar redes LAN que puedan ser capaces de soportar a los usuarios cada vez más demandantes de servicios informáticos a lo que se acceden a través de la red.

Por lo tanto, SDN, no solo está siendo requerida cada vez más por sus bondades en términos técnicos sino también por su capacidad que representa en cuanto a ahorro económico. Según datos extraídos del informe Strategy Analytics del año 2013, en este informe se hacía una proyección para el año 2017, y afirmaban que las redes SDN habrían generado un ahorro de cuatro mil millones de dólares en ese año. El ahorro en este tipo de redes principalmente se refleja en la reducción de cantidad de personal designado a operarlas, por un lado, y, por otro lado, está el ahorro en infraestructura de red al proporcionar seguridad y minimizar el tiempo dedicado a la implantación de cambios(Cision US Inc, 2013).

Si bien es cierto, la tecnología SDN aún no ha sido implantada de manera masiva en Latinoamérica, hay avances en investigación y desarrollo en países como, por ejemplo, Ecuador(Alcívar & Navia, 2020) y Colombia. Sin embargo, se requiere de mayor intervención principalmente de instituciones académicas, en las que se inicie o amplíe la investigación y desarrollo, con la finalidad de impulsar el uso de las redes SDN.

2. CAPÍTULO 2: MARCO TEÓRICO

2.1. Recomendaciones y buenas prácticas de la ONF

La ONF, Open Network Fundation, en su documento de casos de uso y métodos de migración, establece algunas recomendaciones y mejores prácticas para el diseño y migración de una red SDN, entre las más resaltantes para este proyecto se tienen las siguientes:

a) Realizar un estricto análisis de brechas, este análisis permitirá entender a detalle el impacto que tendrá la nueva red en los servicios informáticos existentes. Para cualquier brecha que sea identificada se debe contar con planes alternativos que permitan minimizar los riesgos potenciales que se presenten en la migración.
b) Se debe contar con una detallada lista de verificación previa y posteriores al diseño y migración, estas listas deben ser específicas con muestras y evaluaciones de las aplicaciones y/o servicios de red que serán utilizados para comprobar la conexión y la continuidad del servicio de red. Estas evaluaciones preliminares (antes) y finales ayudan a garantizar que los problemas y fallas relacionados con el diseño y migración de la red sean solucionados a tiempo.
c) Procedimientos de vuelta atrás, es necesario establecer un procedimiento que permita, ante un eventual problema que no se pueda solucionar, regresar la red al estado anterior y no dejarla inoperativa.
d) Analizar todas las características en su conjunto, tener en cuenta las características del protocolo OpenFlow, el controlador y sus capacidades requeridas y los conmutadores de OpenFlow con la finalidad de garantizar y cumplir con los requisitos establecidos para el diseño.
e) Control de versión del protocolo OpenFlow, es importante se verifique muy bien la compatibilidad que existe entre la versión del protocolo OpenFlow que se implementará con la versión del controlador y los conmutadores.
f) Todos los dispositivos OpenFlow deben estar actualizados antes de iniciar el proceso de migración a la red SDN, para ejecutar el código y el firmware de hardware adecuados.
g) Debe existir una confirmación previa de conectividad entre el controlador y los conmutadores OpenFlow.
h) Para solucionar problemas se deben emplear procedimientos de solución de problemas adecuados; por ejemplo, hacer uso del ping, tracert o utilizar alguna aplicación que permita verificar la correcta conexión entre dispositivos.

2.2. Modelos de despliegue de red SDN

Según Pereira & Gamess (2017), existen tres modelos de despliegue SDN, los mismos que se describen a continuación:

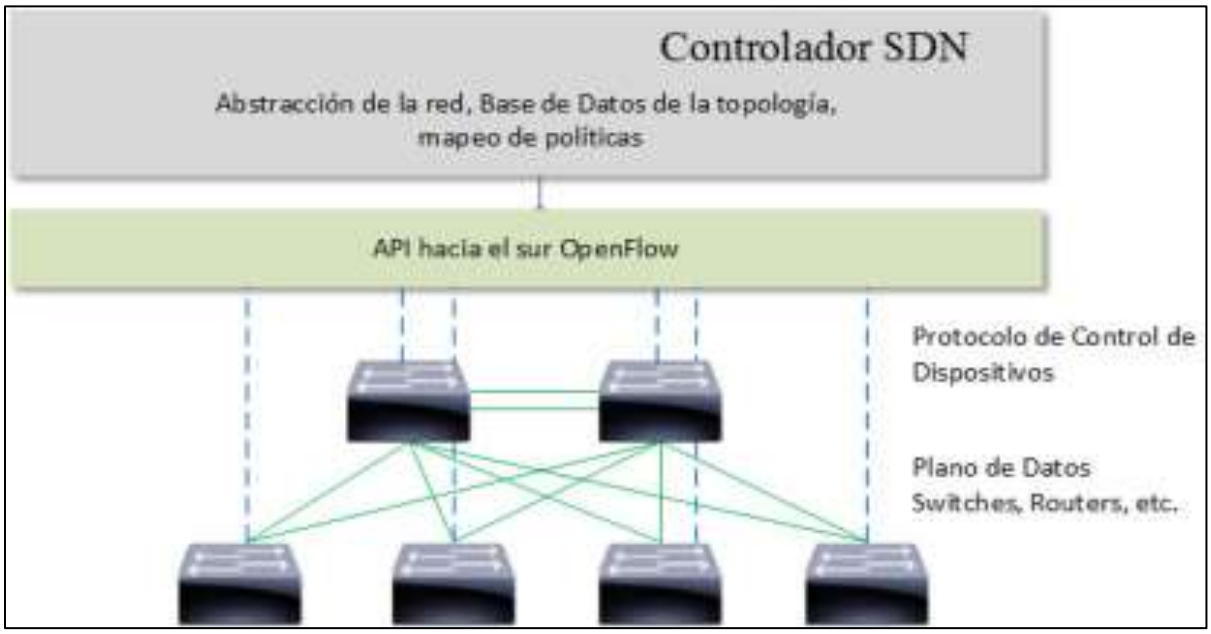
- ✓ Modelo SDN basado en Dispositivos
- ✓ Modelo SDN Overlay
- ✓ Modelo SDN Híbrido

2.2.1. Modelo SDN basado en Dispositivos

El modelo SDN basado en dispositivos es un método de implementación en base a la programación y gestión centralizada de conmutadores de red. En este modelo de despliegue la lógica y políticas de red están definidas en un controlador centralizado el cual es responsable de la configuración y gestión directa de cada conmutador. En la siguiente figura (Figura 12) se puede apreciar este tipo de despliegue compuesto por 6 conmutadores gestionado por un controlador SDN centralizado (Pereira & Gamess, 2017).

Figura 12

Modelo SDN basado en Dispositivos



Nota. De "Lineamientos para el Despliegue de Redes SDN/OpenFlow", por Pereira & Gamess, 2017 (https://www.researchgate.net/publication/333902840_Lineamientos_para_el_Despliegue_de_Redes_SDNOpenFlow).

En este modelo SDN, el controlador es encargado de tomar decisiones como las de establecer políticas de seguridad, enrutamiento, aplicar reglas de calidad y servicio, entre otras funciones enfocadas en la gestión y el control de la red.

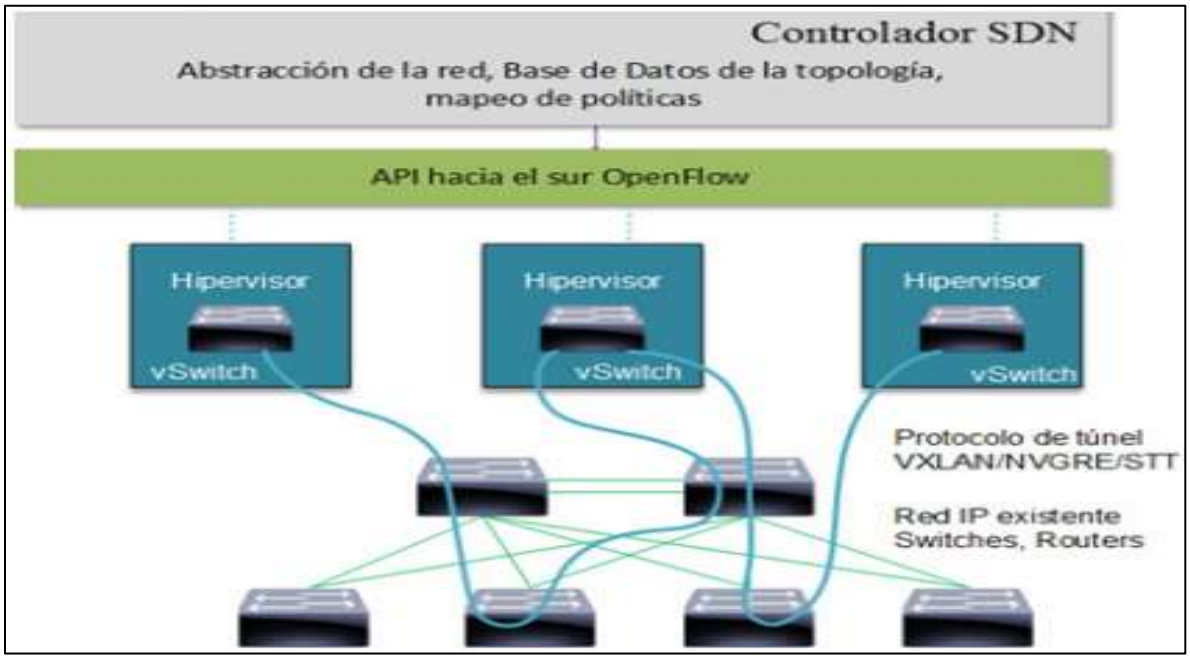
2.2.2. Modelo SDN Overlay

El modelo SDN Overlay es un método de implementación que permite crear una red virtualizada sobre una red existente. En este modelo la capa Overlay se crea mediante software la cual se superpone a la red física subyacente. Es importante resaltar que esta red virtual es independiente de la red física, con lo cual se logra mayor agilidad y flexibilidad dado que los cambios en las configuraciones de la red virtual son de manera asilada a la red física existente y esta última no se verá afectada.

Este modelo generalmente se da en ambientes virtualizados donde los nodos finales son equipos virtuales que integran al hipervisor en ambientes de nube y de Data Center con virtualización de servidores (Pereira & Gamess, 2017).

Figura 13

Modelo SDN Overlay



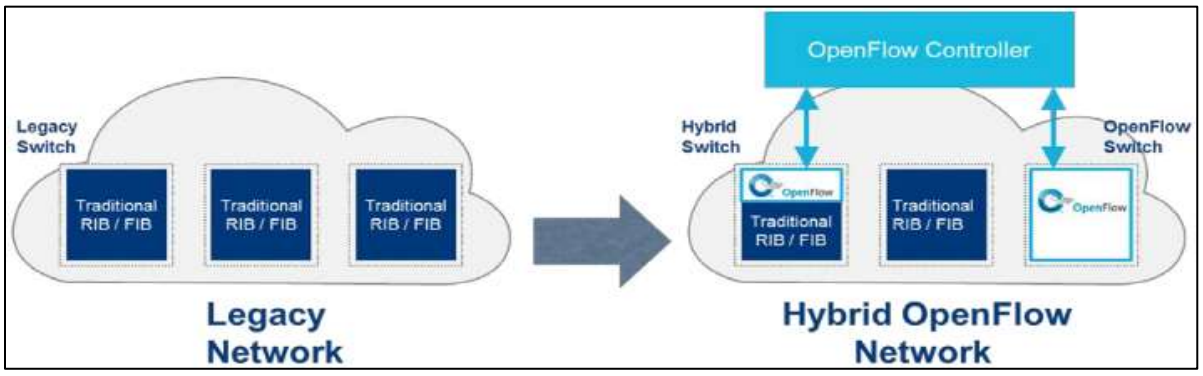
Nota. De "Lineamientos para el Despliegue de Redes SDN/OpenFlow", por Pereira & Gamess, 2017 (https://www.researchgate.net/publication/333902840_Lineamientos_para_el_Despliegue_de_Redes_SDNOpenFlow).

2.2.3. Modelo SDN Híbrido

Este modelo de red SDN es una combinación de los dos modelos descrtos anteriormente, es decir se utilizan tanto conmutadores o enrutadores programables como equipos bajo el control de un controlador central. En este esquema se pueden contar con Gateway que funcionan con protocolos OpenFlow como con protocolos tradicionales de tal manera que este Gateway tendrá la capacidad de comunicarse con los dispositivos SDN, así como con los dispositivos de red tradicionales. (Pereira & Gamess, 2017) Este método toma relevancia en la migración hacia las redes SDN, porque brinda la flexibilidad de convivir con los tipos de arquitectura de red y tener una migración gradual sin la necesidad de hacerlo al 100%.

Figura 14

Modelo de despliegue híbrido



Nota. De "SDN Migration Considerations and Use Cases ONF Solution Brief", por ONF, 2014b (https://opennetworking.org/wp-content/uploads/2014/10/sb-sdn-migration-use-cases.pdf).

2.3. Controlador SDN

El controlador SDN es el elemento esencial en una arquitectura de red SDN. Se trata de un sistema (software) que se ejecuta dentro de una computadora/servidor o máquina virtual y es el que administra y controla la red sin la necesidad de tener la inteligencia de la red distribuida de manera individual en todos los conmutadores y/o enrutadores que lo conforman. El controlador ofrece múltiples servicios a los dispositivos que los gestiona, entre lo más resaltantes tenemos:

a) Descubrimiento de la red y administración de la topología, es el encargado de descubrir los conmutadores, enrutadores y enlaces disponibles en la red y recopilar información

de la topología para en función de ello tomar las decisiones que corresponda como la de envío de paquetes y enrutamiento.

b) Monitoreo y análisis de red, tiene la capacidad de recopilar del rendimiento de la red, estadísticas de tráfico y el estado de los dispositivos y presentar esta información para poder conocer el rendimiento y el estado de la red.

c) Administración de notificaciones, se encarga de la gestión de la comunicación del plano de control a todos los conmutadores o enrutadores de la red. (Pereira & Gamess, 2017)

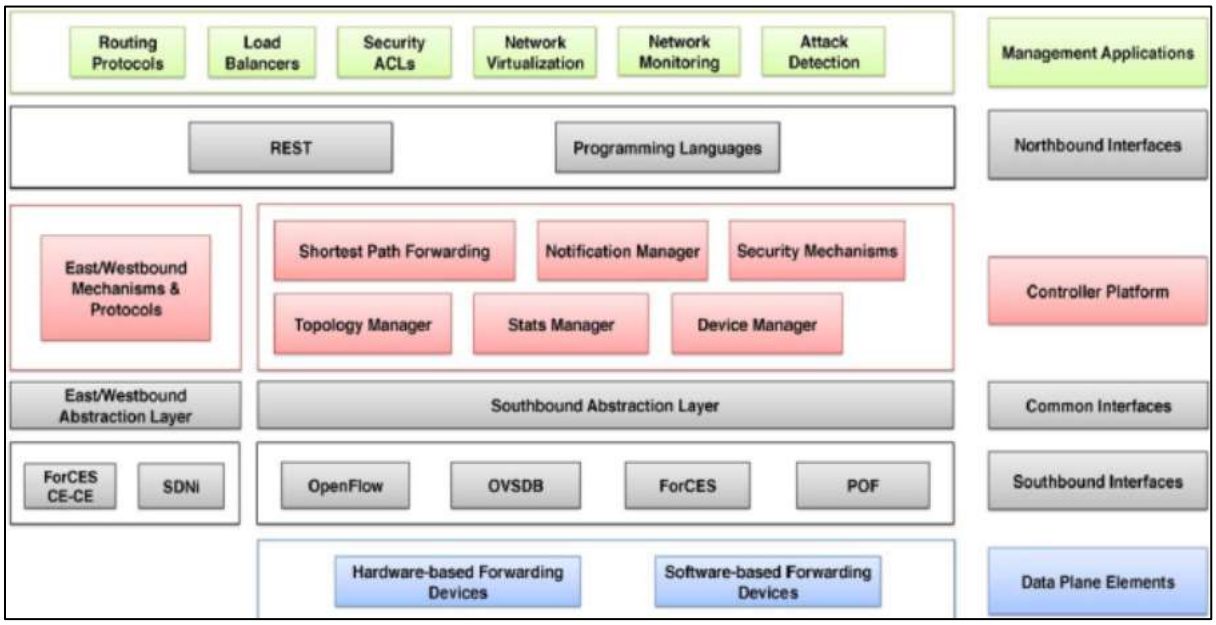
d) Administración de los dispositivos de red, esto lo logra mediante los protocolos de control SDN, como por ejemplo OpenFlow. Mediante estos protocolos y de manera centralizada, el controlador se encarga de programar y configurar los conmutadores y enrutadores de la red.

e) Automatización y programabilidad, está en la capacidad de ofrecer una interfaz programable, generalmente a través de APIs, que permite al personal de administración y desarrollo programar y automatizar tareas de gestión de la red; con esto se logra mayor facilidad en la implementación de políticas de red, configurar los dispositivos y despliegue de servicio. En general, hace que la gestión sea más rápida y eficiente.

f) Mecanismos de seguridad, el controlador se encarga de brindar mecanismos para proteger a la red y mantenerla siempre seguridad (Pereira & Gamess, 2017).

Figura 15

Composición de un Controlador SDN



Nota. De "Lineamientos para el Despliegue de Redes SDN/OpenFlow", por Pereira & Gamess, 2017 (https://www.researchgate.net/publication/333902840_Lineamientos_para_el_Despliegue_de_Redes_SDNOpenFlow).

Según (García et al., 2014) Para decidir entre uno u otro controlador es necesario tener en cuenta lo siguiente:

a) **Requisitos y propósito de la red:** es muy importante conocer los requisitos y el propósito de la red que se gestionará mediante el controlador, también sumará mucho conocer los servicios y aplicaciones que se implementarán, así como la claridad sobre los problemas que se tienen actualmente y se buscar resolver, estos puntos básicos, pero muy importantes, ayudarán a comprender la necesidad que se tiene para elegir el controlador más adecuado.

b) **Compatibilidad con estándares y protocolos:** es indispensable conocer la compatibilidad del controlador con los protocolos e inclusive sus versiones, pues no todas las versiones se pueden adecuar a la necesidad identificada en el punto anterior. Por ejemplo, en nuestro caso es importante ver que el controlador sea compatible con OpenFlow y luego con qué versiones de este es compatible.

c) **Rendimiento/Escalabilidad/Disponibilidad:** Esto es muy relevante puesto que dependerá de la necesidad identificada en el primer punto; además, el controlador debe estar en la capacidad de adecuarse a la complejidad de la red (tamaño o modo de operación) para poder ser escalable en cuanto a brindar un excelente rendimiento para gestionar la cantidad de dispositivos y los flujos de datos con los que se cuenta actualmente y las proyecciones a futuro. En este punto también es importante tomar en cuenta si el controlador es capaz de poder ser implementado para que funcione en alta disponibilidad.

d) **Flexibilidad/Modularidad/Soporte:** Tomar en cuenta la arquitectura del controlador y la capacidad para adaptarse a distintos modelos de red. Si el controlador no es modular y flexible, difícilmente se podrá agregar nuevas funcionalidades y extensiones en base a nuevos requerimientos de la red. Así mismo, es indispensable que el controlador tenga como respaldo una comunidad de usuarios y desarrolladores; sin una comunidad activa será muy complicado poder tener soporte o documentación actualizada o que continúe la contribución para mejoras del controlador.

e) **Nivel de integración:** tener en consideración la capacidad y el nivel de complejidad con el que controlador se integra a otras herramientas existentes en la red, como por ejemplo herramientas de monitoreo, escaneos, virtualización, etc.

f) **Casos de uso/éxito:** esto ayudará a comprender dónde fue utilizado el controlador y que resultados se obtuvieron en otros escenarios reales, inclusive, con esto se podría tener referencia de cómo fueron solucionados ciertos problemas que posiblemente se tengan que afrontar en la red a implementar.

g) **Costos/Factibilidad:** Finalmente y no menos importante está la evaluación económica, se debe considerar el costo total de propiedad (TCO) del controlador SDN, teniendo en cuenta el licenciamiento en caso de ser necesario, mantenimiento, capacitación y cualquier otro recurso de hardware requerido. En base a los resultados, se debe hacer la evaluación de la factibilidad en función con los beneficios y el valor que proporciona el controlador para tu red y la organización.

Entre los controladores de código abierto (open source) existentes actualmente tenemos: ONOS, OpenDayLight, Floodlight, Beacon, NOX/POX, OpenContrail, FlowVisor, Ryu, Atrium, NodeFlow, FlowER, IRIS, etc. Debido a que la lista de controladores es amplia, nos centraremos en 5 de ellos para realizar una comparación entre sus atributos más resaltantes: NOX/POX y Beacon por ser de los primeros controladores SDN en ser desarrollados, Floodlight y OpenDayLight porque son los controladores más utilizados además de contar con el respaldo de grades organizaciones y empresas en el rubro de las telecomunicaciones; y finalmente el controlador ONOS, el que será utilizado en este proyecto.

Figura 16

Comparación de controladores SDN open Source NOX/POX, Beacon, OpenDayLight, Floodlight y ONOS

	NOX/POX	**Beacon**	**Floodlight**	**OpenDayLight**	**ONOS**
Última Versión	0.2.3	1.2	1.2	12 (Magnesium)	2.0
Soporte OpenFlow	v1.0	v1.0	v1.0 - v1.5	v1.0 - v1.3	v1.0 - v1.5
Virtualización	Mininet y Open vConmutador	Mininet y Open vConmutador	Mininet y Open vConmutador	Mininet y Open vConmutador	Mininet y Open vConmutador
Lenguaje de Programación	C++	Java	Java	Java	Java
Provee REST API	No	No	Si	Si	Si
Interfaz Gráfica	Python+, QT4	Web	Web	Web	Web
Soporte de Plataformas	Linux	Linux, Mac OS, Windows y Android para móviles	Linux, Mac OS, Windows	Linux, Mac OS, Windows	Linux, Mac OS, Windows
Soporte de OpenStack	No	No	Si	Si	Si
Multiprocesos	Si	Si	Si	Si	Si
Código Abierto	Si	Si	Si	Si	Si
Año de Lanzamiento	2008	2010	2014	2013	2014
Integración recursos virtualizados y reales	Si	Si	Si	Si	Si
Documentación	Media	Buena	Buena	Media	Buena

A continuación, nos centraremos en describir lo más importante y resaltante del controlador ONOS, el cual será utilizado en el desarrollo de este proyecto.

2.3.1. Controlador Open Network Operating System (ONOS)

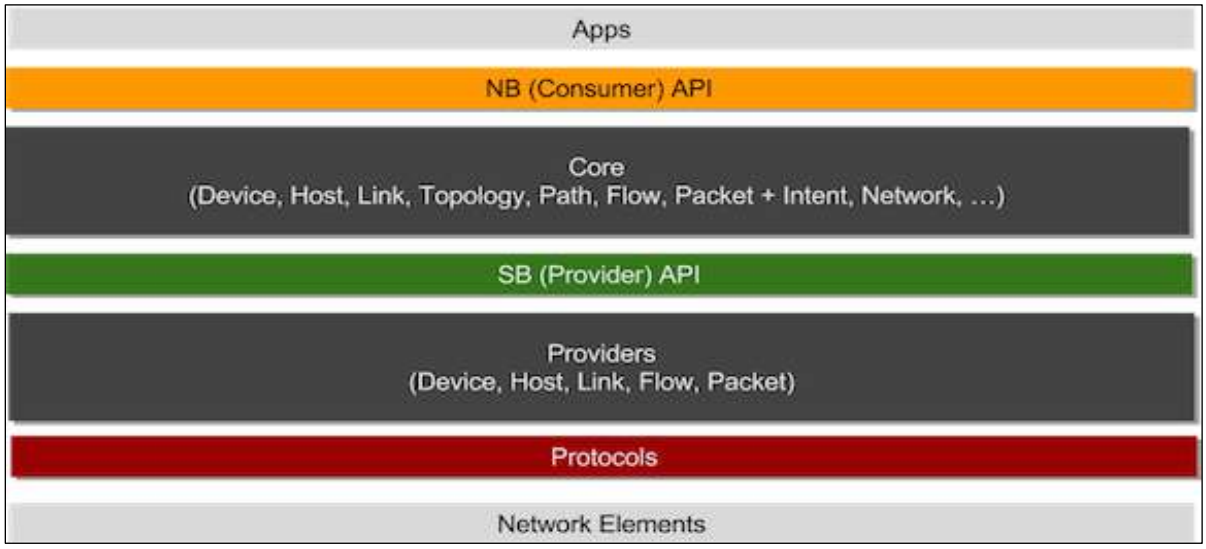
a) Características

ONOS (Open Network Operating System), es un proyecto de código abierto que desde el año 2015 forma parte de The Linux Foundation. Sin embargo, este controlador surgió en el año 2014 gracias a Open Networking Lab. Entre las principales características que ofrece este controlador SDN, podemos mencionar las siguientes:

- ✓ **Alta disponibilidad**: ONOS sostiene las redes de los operadores más rigurosos y garantiza la precisión de las conexiones para que todos sus usuarios puedan tener una excelente experiencia sin percibir pérdida de conexiones o inestabilidad de la red.
- ✓ **Rendimiento a escala**: El controlador se adapta rápidamente al crecimiento de la red y soporta una gran cantidad de solicitudes; estas pueden ser millones, y las puede atender en tiempos menores a 50 milisegundos. Para esto, ONOS tiene la capacidad de agregar nuevas instancias del controlador de manera automática cuando necesita más capacidad en el plano de control.
- ✓ **Independencia y soporte de protocolos:** ONOS proporciona APIs y es compatible con varios protocolos y estándares del medio, esto sin duda facilita el que el controlador se pueda integrar con facilidad y permita la interoperabilidad con diferentes soluciones. Como ejemplo principal tenemos a NetFlow, pero también hay otras como, REST API, BGP-LS, NETCONF, gNMI, etc.
- ✓ **Aplicaciones escritas en lenguaje de alto nivel:** Estas aplicaciones son escritas en código Java. Sin embargo, existe también una API (ONOS Java API) con una amplia gama de recursos disponibles para el desarrollo de aplicaciones.(Onosproject, 2020)
- ✓ **Balanceo de carga:** Mediante la funcionalidad de alta disponibilidad que ONOS ofrece, se puede realizar balanceo de carga; esto, con el fin de no sobrecargar únicamente a un controlador y optimizar el rendimiento de todos los controladores que están en alta disponibilidad.

Figura 17

Arquitectura del controlador ONOS



Nota. De "ONOS Java API (1.13.10), por Onosproject, 2020 (http://api.onosproject.org/1.13.10/).

Tal como se puede observar en la figura 17, en la arquitectura de ONOS se distingue de las capas de aplicación, control y datos.

Las Southbound APIS: En un módulo que interactúa con la red.

El núcleo del sistema: es en donde se recibe y procesa la información del estado de la red.

Northbound APIS: Permiten que aplicaciones, sistemas u servicios en los niveles superiores de la arquitectura de la red se comuniquen, utilicen o puedan acceder a las capacidades del controlador y a la red que este administra.

b) Eventos e Intents

El controlador ONOS se comunica con las aplicaciones a través de eventos asíncronos. Para tal fin, existen unas interfaces llamadas Listener que son las que tienen por función capturar los eventos del servicio que se requiera. Estas deben implementarse para indicar cómo tratar los eventos en cuestión.

Así mismo, las aplicaciones se comunican con el controlador para agregar, cambiar o eliminar las entradas en las tablas de flujos. Esta comunicación se puede realizar de dos maneras:

- ✓ **Comunicación directa de la orden:** La aplicación notifica al controlador de los cambios que desea realizar en una entrada en particular.

- ✓ **Comunicación a través de intents:** El Intent Framework es un marco que permite que las aplicaciones especifiquen sus requerimientos de control de red. En tal sentido, el controlador recibe indicaciones de la aplicación en base a lo que esta necesita. Por ejemplo, un intents se puede expresar de la siguiente manera: ***"Necesito comunicar X con Y en base a los N requisitos"***. En este caso, ONOS se encarga de hacer la traducción este requerimiento de intents en entradas en las tablas de flujos y luego transferir a los conmutadores correspondientes.

c) Formas de acceso y gestión

Los administradores de red pueden interactuar con el controlador a través de cualquiera de las tres maneras:

Interfaz Gráfica: ONOS cuenta con una interfaz web (GUI) a la que se puede acceder desde un navegador web. Esta interfaz permite ver la topología de la red, inventario e información de todos los conmutadores y host que pertenecen a la red, las aplicaciones que están instaladas, instalar aplicaciones nuevas. También se puede analizar el tráfico en tiempo real que cursa por cada enlace.

Terminal CLI: Mediante el terminal de línea de comandos se pueden configurar las aplicaciones y conseguir información de la red. Así mismo, brinda una gran cantidad de comandos para interactuar con los dispositivos y aplicaciones y poder realizar diversas configuraciones.

API REST: Permite trabajar las aplicaciones, es decir, realizar la instalación, activación, eliminación, modificación, etc. Así como extraer información de la red, actualizarla y transferir nuevas órdenes y entradas en las tablas de flujos de los conmutadores. Esta interfaz es de bastante utilidad cuando se realizan pruebas.

2.4. Protocolo OpenFlow

OpenFlow es un protocolo de comunicación que se utiliza ampliamente en entornos de redes SDN para separar el plano de datos del plano de control de la red. Este protocolo fue desarrollado por investigadores de la Universidad de Stanford y Berkeley. La primera implementación de referencia fue publicada en el mes de noviembre del 2007 bajo la versión 0.1.0, en el 2008 se hicieron otras publicaciones de las siguientes versiones aún experimentales (0.2.0, 0.8.0, 0.8.1, 0.8.2, 0.8.9) y en el 2009 se publicó la versión 0.9.

El protocolo OpenFlow está basado en una arquitectura cliente-servidor, en la cual el o los controladores de red sirven como servidores y los conmutadores actúan como los clientes. Los conmutadores OpenFlow son dispositivos compatibles con el protocolo que cuenta con una tabla de flujo programable y la capacidad de recibir instrucciones del controlador.

El protocolo brinda una manera estándar de comunicación entre los controladores y los conmutadores, lo cual facilita la interoperabilidad entre distintos dispositivos y soluciones de redes definidas por software.

2.4.1. Versiones

a) Versión 1.0

A finales del año 2009 se publicó la versión 1.0, esta es versión más adoptada, la cual tiene en cuenta los siguientes campos de encabezado que se encuentran en los paquetes Ethernet.

Tabla 3

Campos de cabecera de la versión 1.0

Campos de cabecera Protocolo Open Flow v1.0
Puerto Entrante
Dirección MAC Origen
Dirección MAC Destino
Tipo Ethernet
Id VLAN
Prioridad VLAN
IP Origen
IP Destino
Protocolo IP
Bits ToS (tipo de servicio) IP
Puerto TCP/UDP origen
Puerto TCP/UDP destino

Nota. De "OpenFlow Switch Specification", por Onosproject, 2020 (https://opennetworking.org/wp-content/uploads/2013/04/openflow-spec-v1.0.0.pdf).

La versión 1.0 es la primera versión no experimental del protocolo. Este primer lanzamiento sentó las bases para el protocolo OpenFlow así como para el desarrollo y la adopción de redes definidas por software.

Entre las principales características de esta versión están las siguientes:

- Tablas de flujo: En esta versión se inicia el concepto de este tipo de tablas que las usan los conmutadores para tomar decisiones para el manejo de los paquetes de red. Las entradas de la tabla de flujo constan de campos de coincidencia y acciones que se aplicarán a los paquetes que coincidan con esos campos.
- Campos de coincidencias: Utiliza campos de coincidencia básico. Estos permiten hacer coincidir los campos de encabezado en paquetes como la dirección MAC, la dirección IP, el puerto TCP/UDP, etiqueta VLAN, etc. Estos campos además utilizan para determinar cuáles entradas en la tabla de flujo deben aplicarse a un paquete específico.
- Tabla de acciones: Se definen un grupo de acciones que se podrían aplicar a los paquetes como, por ejemplo, el reenvío de un paquete a un determinado puerto, la modificación del campo de encabezado de un paquete, el envío de un paquete al controlador y más.
- Controlador de red centralizado: Es el actor principal para la toma de decisiones sobre la manera en la que se manejarán los paquetes en los conmutadores; el controlador puede enviar indicaciones a los conmutadores para que estos ejecuten acciones específicas en los paquetes y tomen decisiones para enrutar y gestionar el tráfico.
- Mensajes OpenFlow: Entre los principales mensajes definidos, se tienen a los mensajes de respuesta, de solicitud y de notificación. Estos son utilizados para comunicar a los conmutadores y el controlador de la red.
- Limitaciones: Puesto que se trata de la primera versión, es lógico encontrar algunas limitaciones respecto a las versiones posteriores del protocolo, entre las cuales se tienen a la carencia de soporte y características avanzadas para la gestión de calidad y servicio, agregación de enlaces o enrutamiento por múltiples rutas.

La versión 1.0 del protocolo afirmó las bases para nuevas versiones en las que se introducen mejoras y nuevas funcionalidades; esta versión ha sido sucedida por versiones más recientes como la OpenFlow 1.3 y 1.4 en las que se han incrementado las capacidades de las redes definidas por software.

b) Versión 1.1

Esta versión del protocolo fue publicada en febrero del 2011. En la versión 1.1, a comparación de la versión anterior se agregan mejoras, principalmente en control y gestión de las redes definidas por software. Entre las principales tenemos las siguientes:

- Soporte para múltiples tablas: se introduce el concepto de múltiples tablas de flujo en los conmutadores lo cual brinda una mayor flexibilidad en el procesamiento de paquetes al poder procesarse en varias etapas mediante diferentes tablas. Cada una de las tablas pueden tener sus propias entradas de flujo de manera independientes manteniendo su propia lógica de acciones y coincidencias.
- Coincidencia de campos extendida: Esta versión amplió el conjunto de campos compatibles. A los campos básico de la versión 1.0 se agregaron los campos tipo de servicio (TLS), etiquetas de conmutación de etiquetas multiprotocolo (MPLS) y campos de protocolos específicos.
- Acciones mejoradas: ÓpenFlow 1.1 mejoró las acciones existentes e introdujo nuevas como, cambio de redes LAN virtuales (VLAN), encolamiento de paquetes, el conjunto de campo de encabezado de transporte y el grupo de acciones. Con estas mejoras se logró que los paquetes sean mejor manipulados y controlados en los conmutadores.
- Estadísticas mejoradas: Brindó una mejor capacidad de recopilación de estadísticas en los conmutadores. Estas mejoras permiten que el controlador obtenga mayor información del rendimiento de la red y los flujos de tráfico y de esta manera facilitar la toma de decisiones y poder optimizar la red.
- Mejoras en el soporte de seguridad: incorporó mejoras en esta funcionalidad mediante mecanismos para autenticar y asegurar la comunicación entre el controlador y los conmutadores; con esto se logró mejorar la protección de la integridad y la confidencialidad en los mensajes OpenFlow.

c) Versión 1.2

La versión 1.2 fue publicada en diciembre de 2011. Esta versión adición varias características respecto a las versiones anteriores, entre las más importantes están: la posibilidad de que los conmutadores se conecten a más de un controlador de manera simultánea lo cual permitió robustecer la gestión y el control de la red, también soporta la versión 6 del protocolo IP, el balanceo de carga, esta última característica es muy importante porque permite la posibilidad de que la red esté preparada ante fallos.

d) Versión 1.3

La versión 1.3 del protocolo fue publicada e junio del 2012, como era de esperarse, esta versión incluye nuevas y mejoradas características respecto a las versiones que le

anteceden, entre la cuales están principalmente la de contar con la posibilidad de controlar mediante medidores de flujo la tasa de paquetes, en coincidencia de campos extendida se incluyen nuevos campos para el protocolo IP versión 6, para calidad y servicio entre otros. En esta versión, también, se introduce el concepto de grupos de tablas, lo cual permite agrupar varias tablas de flujo y aplicar acciones en conjunto a ellas.

OpenFlow 1.3 introduce además una nueva tabla llamada "Tabla de medidores" para ampliar la capacidad de QoS; la tabla del medidor consta de entradas de medidor identificadas por el identificador del medidor. Además, cada entrada en la tabla del medidor contiene una lista de "Bandas del medidor", que especifican la tasa y el comportamiento (eliminar o comentar DSCP). Cuando un paquete coincide con una entrada de medidor, se aplicará la banda del medidor con la tasa configurada más alta que sea menor que la tasa medida actual, por lo que es compatible con el modelo DiffServ. Además, en esta versión se ha ampliado la tabla de flujo con una entrada de tabla perdida (Open Networking Foundation, 2014a). En la versión anterior de OpenFlow, un paquete se descartaba o se enviaba al controlador en un mensaje de entrada de paquetes; con la entrada de tabla perdida, el comportamiento de procesamiento de los paquetes no coincidentes sería más flexible que el de la versión anterior (Ching-Hao & Ying-Dar, 2015).

e) Versión 1.4

Esta versión del protocolo fue publicada en marzo del 2013. Tal como en las versiones anteriores, en esta versión se continuó afinando y robusteciendo el protocolo. Entre las principales características mejoradas o nuevas en esta versión están: la mejora en las tablas de grupos, en estas se agregaron nuevos tipos como el grupos select y grupos indirect con los cuales se logra mayor flexibilidad en la gestión de los flujos; se mejora la funcionalidad de calidad y servicio mediante mecanismos para clasificar y controlar el tráfico en base a sus prioridades y requisitos de calidad y servicio, también se incluye el soporte para puertos o interfaces de fibra óptica (Open Networking Foundation, 2013).

f) Versión 1.5

Esta versión del protocolo 1.5 fue publicada en diciembre de 2014, introduce tablas de salida con lo cual se puede realizar el procesamiento en el contexto del puerto de salida, también se agregan canalización consciente del tipo de paquete para pauetes diferentes a los ethernet, como por ejemplo IP, protocolo punto a punto (PPP); coincidencia de banderas TCP mediante un nuevo campo para que coincida con los bits de bandera den el

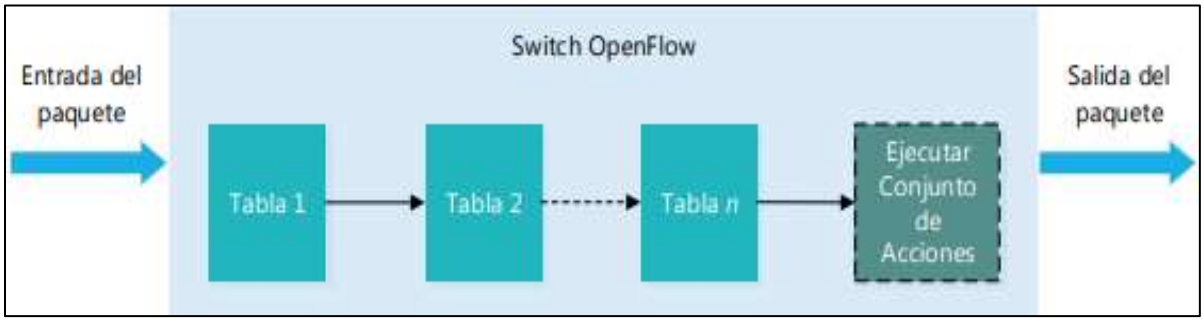
encabezado, este campo brinda la posibilidad de hacer coincidir todos los indicadores SYN, ACK y FIN; entre otras características (Open Networking Foundation, 2014).

2.4.2. Tablas OpenFlow

En OpenFlow, no se tiene una cantidad definida de tablas. Sin embargo, en una configuración típica de conmutadores compatibles, o de manera general cualquier dispositivo compatible con OpenFlow, se pueden considerar 4 tipos de tablas.

Figura 18

Paquete de datos a través del pipeline de procesamiento



Nota. Adaptado de "OpenFlow Switch Specification Version 1.5.1 (Protocol version 0x06)", por ONF, 2015 (http://www.opennetworking.org).

a) Tablas de Coincidencias:

La tabla de coincidencia en un dispositivo OpenFlow, es la primera tabla que un paquete debe cruzar o afrontar. La función de esta tabla es hacer coincidir los campos del encabezado del paquete con los valores detallados en las entradas de la tabla. Estas coincidencias serán determinantes para el manejo que se le dará al paquete.

Figura 19

Tabla de coincidencias detallada

Campo	Descripción	Máscara	Bits	Prerrequisito
IN_PORT	Puerto de ingreso. Puede ser un puerto físico o uno lógico definido por el swtich	No	32	Ninguno
ETH_DST	Dirección destino Ethernet. Puede usar una máscara de bits arbitraria	Si	48	Ninguno
ETH_SRC	Dirección fuente Ethernet. Puede usar una máscara de bits arbitraria	Si	48	Ninguno
ETH_TYPE	Tipo Ethernet de la carga de datos del paquete.	No	16	Ninguno
IP_PROTO	Número de protocolo IPv4 ó IPv6	No	8	ETH TYPE=0x0800 ó ETH TYPE=0x86dd
IPV4_SRC	Dirección fuente IPv4. Puede usar una máscara de bits arbitraria o de subred	Si	32	ETH TYPE=0x0800
IPV4_DST	Dirección destino IPv4. Puede usar una máscara de bits arbitraria o de subred	Si	32	ETH TYPE=0x0800
IPV6_SRC	Dirección fuente IPv6. Puede usar una máscara de bits arbitraria o de subred	Si	128	ETH TYPE=0x86dd
IPV6_DST	Dirección destino IPv6. Puede usar una máscara de bits arbitraria o de subred	Si	128	ETH TYPE=0x86dd
TCP_SRC	Puerto TCP fuente	No	16	IP PROTO=6
TCP_DST	Puerto TCP destino	No	16	IP PROTO=6
UDP_SRC	Puerto UDP fuente	No	16	IP PROTO=17
UDP_DST	Puerto UDP destino	No	16	IP PROTO=17

b) Tabla de Flujo

Las tablas de flujo almacenan las reglas de flujo que serán aplicados a los paquetes en base la información identificada en la tabla de coincidencias. En cada una de las reglas se tienen definidas acciones tales como actualización de encabezados, envío a otro puerto, etc.

Figura 20

Componentes principales de una entrada de tabla de flujo del protocolo Open Flow

Campos de Coincidencia	Prioridad	Contadores	Instrucciones	Tiempos de Espera	Cookie

Nota. Adaptado de "OpenFlow Switch Specification Version 1.5.1 (Protocol version 0x06)", por ONF, 2015 (http://www.opennetworking.org).

En la figura 20, se muestra la entrada de tabla de flujo, en la cual están los principales componentes.

Cada entrada de tabla de flujo contiene los siguientes elementos:

- **Campos de coincidencias:** Se utilizan para encontrar coincidencias con los paquetes, están formados por el puerto de origen y los encabezados.
- **Prioridad:** Es la precedencia que tiene la entrada de tabla de flujo (formadas por campos, contadores y múltiples instrucciones) dentro de la tabla de flujo.
- **Contadores:** Estos se actualizan al encontrar coincidencias con los paquetes.
- **Instrucciones:** Son utilizadas para cambiar el conjunto de acciones o el procesamiento en pipeline. Así mismo, estas definen el conjunto de acciones que se relizarán sobre el paquete.
- **Tiempos de espera:** Aquí se define el tiempo máximo fuera de actividad antes de que la entrada de tabla de flujo sea expirada por el conmutador.
- **Cookie:** Los controladores pueden usarlo para filtrar estadísticas, modificar o eliminar entradas de tabla de flujo, pero no son utilizados al momento de procesar los paquetes.

c) Tablas de Grupo

La tabla de grupo es utilizada para agrupar más de un puerto o flujo en una única entidad de grupo. Además, permite la posibilidad de ejecutar acciones como el reenvío a más de un puerto o inclusive clonar paquetes a diversos destinos.

d) Tabla de Acción

La tabla de acción determina las acciones que se pueden tomar en base a los resultados de las entradas en la tabla de flujo. Se pueden tener acciones para un paquete como las de reenviarlo a un puerto determinado, descartarlo, modificar los encabezados, registrarlo, entre otras.

a) **Tabla de Metadatos**

La tabla de metadatos es utilizada para guardar metadatos adicionales relacionados con un flujo. Estos metadatos son valores que se los puede utilizar el controlador u otras tablas de OpenFlow para tomar decisiones en cuanto al manejo de los paquetes.

2.4.3. Canal Seguro Openflow

De acuerdo con (Open Networking Foundation, 2015), este canal es el medio físico por donde se envía el tráfico de control entre el controlador y los conmutadores y puede consistir en una red dedicada o utilizar la misma infraestructura que la red de tráfico de datos: Hay dos formas de hacerlo, la primera es control fuera de banda (Out Of Band Control) y la segunda, control en banda (In Band Control)

a) Control fuera de banda

En el control fuera de banda se utilizan puertos o interfaces Ethernet además de enlaces aislados, es decir la red utilizadas es completamente diferente para la conexión de los conmutadores al controlador. La separación de la infraestructura de red puede ser a nivel físico o lógico. En la infraestructura física, los conmutadores OpenFlow cuentan con un puerto de "administración" físico, el mismo que debe estar conectado al controlador. En la infraestructura lógica, se usan túneles o redes lógicas independiente sobre la misma infraestructura física, como ejemplo de esta implementación podemos citar al documento de la ONF (Casos de Uso y Métodos de Migración - Migration Use Cases and Methods), en este caso se menciona que en el despliegue original de Openflow en Stanford, los conmutadores fueron configurados con 3 VLANs, la primera se utilizó para el tráfic de control Openflow, la segunda fue utilizada para experimento de tráfico de datos con Openflow, y la última fue utilizada para el tráfico productivo (Open Networking Foundation, 2015).

b) Control en Banda

El control en banda hace uso de los propios enlaces de la red, es decir, la red tanto para el tráfico de control como para el tráfico de datos. Es importante mencionar que para este tipo de control, se requiere generalmente que conmutadores cuenten con un conjunto previamente definido de reglas para conectarse con el controlador (Open Networking Foundation, 2015).

2.5. Emulador Mininet

Mininet es una plataforma de pruebas de red de código abierto, que se puede configurar de una manera muy rápida. Es la herramienta que más se conoce que brinda apoyo a la investigación de las redes definidas por software OpenFlow. Usa hosts virtualizados, conmutadores OpenFlow y enlaces para la creación de una red dentro de un mismo núcleo del sistema operativo. Así mismo, utiliza los mismos mecanismos de una red real para el procesamiento de paquetes y de esta manera conectarse a una red real.

Una aplicación de controlador real OpenFlow como por ejemplo el utilizado en este proyecto (OpendayLight), en una red OpenFlow emulada por Mininet, Se puede ejecutar en una computadora remota o en la misma computadora emulando el host virtual.

Mininet permite la instalación, principalmente de 2 formas, la primera de ellas es mediante la descarga de una imagen de máquina virtual, con la herramienta ya instalada, la cual puede ser ejecutada luego sobre un software de virtualización como por ejemplo VirtualBox, la segunda forma es la instalación nativa sobre un sistema operativo Linux Ubuntu (recomendado) o Fedora (Mininet Project Contributors, 2018)

3. CAPÍTULO 3: ANÁLISIS DEL PROBLEMA

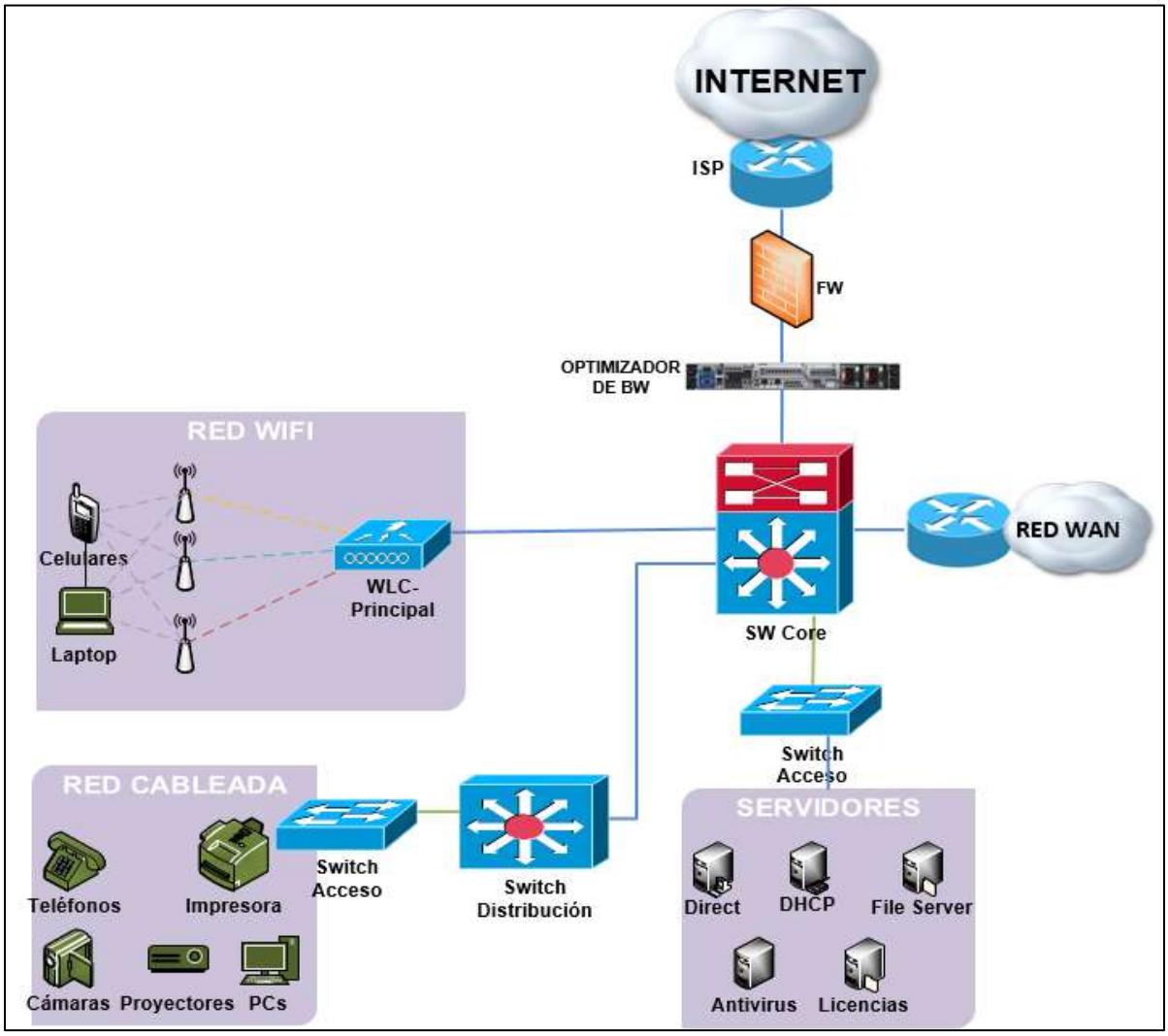
3.1. Situación Actual

Las redes LAN de las universidades, cuentan con una cantidad bastante grande de conmutadores en cada uno de sus campus universitarios. Esto, debido a la cantidad de ambientes académicos (Aulas y laboratorios), centro de información y ambientes administrativos a los cuales se les tiene que brindar conectividad mediante una red cableada.

- Las redes LAN en un campus universitario, objeto de estudio, están basadas en una arquitectura jerárquica tradicional de tres capas (Acceso, Distribución y Core) tal como se muestra en el diagrama de la *figura 21*.

Figura 21

Diagrama general de la Red de un Campus Universitario



Así mismo, la red del campus universitario, que se representa en el diagrama de *la figura 21*, se caracteriza por lo siguiente:

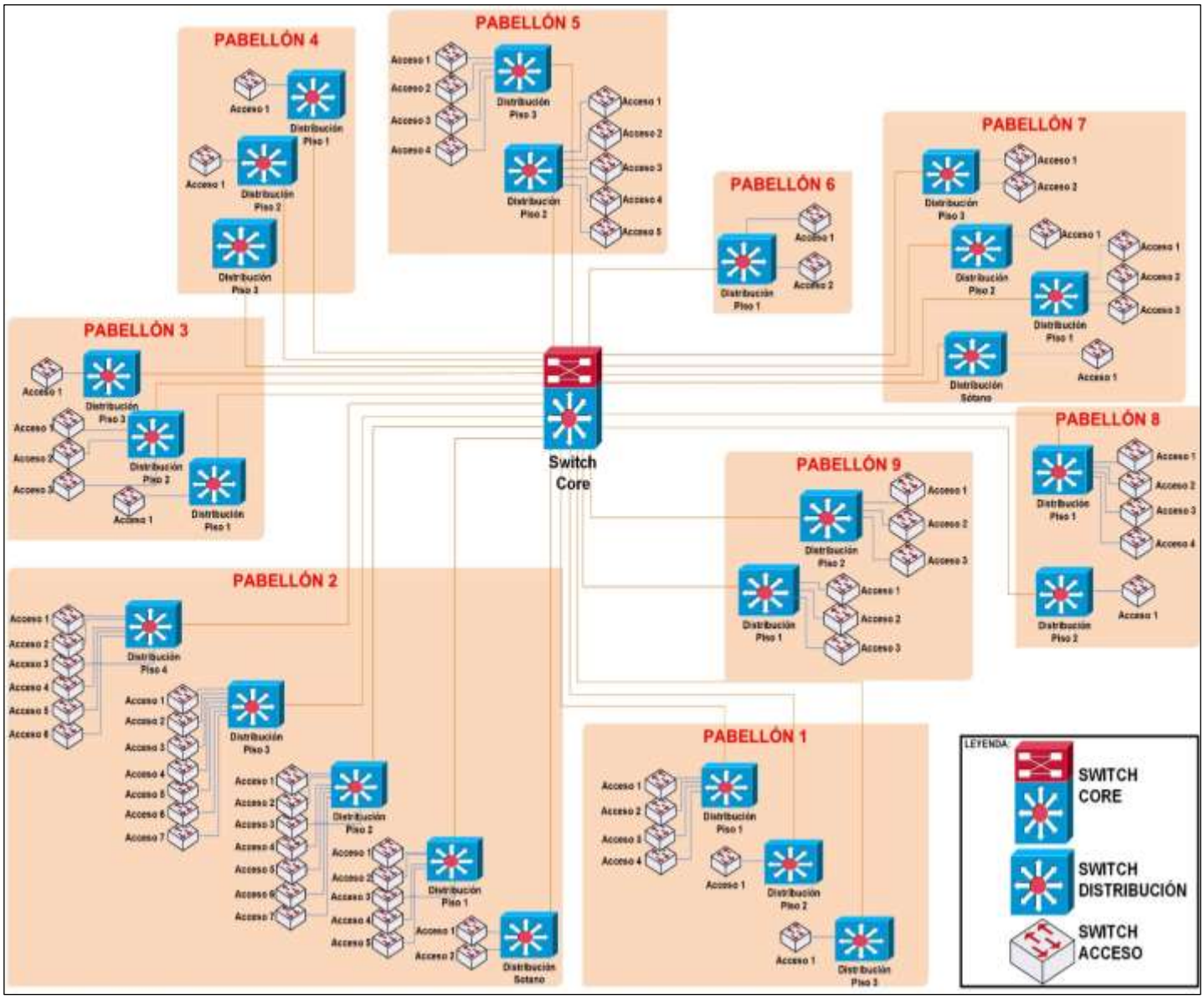
- **Enlace a Internet,** este enlace a internet, suministrado por un Proveedor de Servicios de Internet (ISP), considerado uno de los servicios de TI más importantes, es la base para el acceso a muchos servicios informáticos que las universidades tienen alojados en Centro de Datos de proveedores externos.
- **Firewall,** equipo principal de seguridad informática, brinda la seguridad perimetral desde la internet hacia la red del campus universitario y viceversa.
- **Equipo Optimizador de Ancho de Banda,** equipo mediante el cual se pueden establecer políticas para la optimización de ancho de banda a nivel de redes, direcciones IP, protocolos y aplicaciones; con el propósito de tener un mejor control del ancho de banda de internet contratado.
- **Conmutador Core**, estos equipos están ubicados en el nivel superior de la jerarquía, los mismo que tienen por finalidad la de concentrar a los conmutadores del nivel intermedio, los controladores WiFi y los ruteadores del ISP para el enlace a internet o enlaces WAN.
- **Conmutadores de distribución,** estos equipos están ubicados en el nivel intermedio de la jerarquía, los mismo que tienen como finalidad la de concentrar a los conmutadores del nivel de acceso e interconectar con el nivel superior, es decir con el nivel de Core.
- **Conmutadores de acceso**, estos equipos están ubicados en el nivel más bajo de la jerarquía, los mismo que tiene como función concentrar a los dispositivos finales; que pueden ser: impresoras, teléfonos, cámaras IP, computadoras, etc.
- **Servidores,** equipo en los cuales se implementan múltiples aplicaciones y/o servicios informáticos que se tienen de manera local. Por ejemplo, Directorio Activo, DNS, DHCP, Base de Datos, entre otros.
- **Wireless Lan Controller (WLC),** este equipo es el que gestiona todas las antenas inalámbricas (WiFi) del campus universitario, las mismas que sirven para dar el servicio WiFi a los estudiantes, docentes y personal administrativo.
- **Red de Área Amplia (WAN),** red que sirve para la comunicación entre los campus universitarios adicionales o sedes remotas con las que cuentan las universidades.

ESCENARIO DEL PROYECTO

El escenario sobre el cual se plantea el presente proyecto se representa en el diagrama de red de la *figura 22*. En este diagrama se describe la gran cantidad de conmutadores que un campus universitario puede llegar a tener, los mismos que no siempre están gestionados de manera centralizada; y si lo estuvieran, necesariamente se tendría que depender de una herramienta de gestión para una marca o fabricante determinado; es decir, todos los conmutadores y la herramienta de gestión tienen que pertenecer a un mismo fabricante para mantener la compatibilidad entre sí.

Figura 22

Diagrama de Red LAN referencial de un Campus Universitario



En este diagrama se detalla la manera en que están conectados los conmutadores de la capa de Acceso, Distribución y Core bajo la arquitectura de red clásica o tradicional. Como se puede observar, las redes LAN de un campus universitario podrían llegar a ser muy densas. Los conmutadores del escenario planteado están divididos en tres diferentes fabricantes (CISCO 50%, JUNIPER 30% y DELL 20%).

En este sentido, el escenario sobre el cual está planteado el presente proyecto contempla lo siguiente:

- **Pabellones,** el escenario contempla un campus universitario con 9 pabellones, los mismos que pueden llegar a concentrar a más de 20 mil personas entre estudiantes, docentes y personal administrativo, en 9 pabellones están distribuidos las aulas teóricas, laboratorios de cómputo, laboratorios especializados (electrónica, redes, arquitectura, etc.), Biblioteca, oficinas administrativas, etc.
- **Conmutadores,** entre conmutadores de distribución y acceso, según la arquitectura jerárquica tradicional, el campus universitario contemplado en el proyecto está compuesto de 100 conmutadores para brindar conectividad de red a todos los pabellones.
- **Redes LAN Virtuales (VLANs),** para lograr la segmentación de la red tal como se describe a continuación, es necesario se haga mediante la creación de redes LAN virtuales (VLAN). La cantidad podría llegar a superar las 70 VLANs:
  - ✓ **VLANs WiFi Alumnos,** con el fin de tener dominios de broadcast no muy extensos, en cuanto a dispositivos conectado en una misma red, para la red inalámbrica de estudiantes se crean múltiples VLANs para brindar una conexión de red óptima a los estudiantes.
  - ✓ **VLANs WiFi Docentes,** Dependiendo de la cantidad de docentes, se puede llegar a tener hasta 3 VLANs con diferentes segmentos de red.
  - ✓ **VLANs WIFI Administrativos,** Dependiendo de la cantidad de personal administrativo, se puede llegar a tener hasta 2 VLANs con diferentes segmentos de red.
  - ✓ **VLANs WiFi Invitados,** para brindar una conexión WiFi con políticas de acceso específicas
  - ✓ **VLANs para Laboratorios de cómputo,** estos laboratorios se encuentran en diferentes pabellones
  - ✓ **VLANs para Laboratorios especializados,** estos laboratorios se encuentran en diferentes pabellones

- ✓ **VLAN para Biblioteca,** una VLAN para concentrar a todas las computadoras que exista en la biblioteca, la misma que a su vez se puede encontrar dividida en distintos pabellones.
- ✓ **VLANs para Administrativos (cableada),** existentes para los equipos de cómputo (computadoras de escritorio, laptops) que se conectan a la red mediante cable.
- ✓ **VLAN para Proyectores,** para gestionar los proyectores mediante IP.
- ✓ **VLAN para Aulas Teóricas,** para conectar las Computadoras de las aulas teóricas a la red
- ✓ **VLAN para Servidores,** red dedicada únicamente a los servidores que están en el Centro de Datos e implementar políticas de accedo.
- ✓ **VLAN para Gestión de Servidores,** Red dedicada para los puertos de gestión de los servidores.
- ✓ **VLAN para Gestión de conmutadores,** red para la gestión de los conmutadores existentes en el campus.
- ✓ **VLAN para Gestión de Antenas WiFi,** red dedicada para la gestión de las antenas WIFI, las que podrían llegar a ser más 400.
- ✓ **VLAN para Telefonía,** red dedicada para los equipos de telefonías (teléfonos, centrales, etc.
- ✓ **VLAN para Impresoras,** red para todas las impresoras de red.

- **Listas de Control de Acceso (ACLs),** están implementadas con el fin de aplicar políticas de acceso o restricción entre las diferentes VLANs o divisiones en la red existentes.

3.2. Análisis del Problema.

En función del escenario y basado en mi experiencia personal, la red LAN de un campus universitario tal como se ha planteado; con una gran cantidad de conmutadores, VLANs y políticas de acceso, la gestión de los conmutadores se torna muy compleja, e implica el desarrollo de una serie de actividades. A continuación, mencionaremos algunas de las actividades más relevantes tomadas en cuenta; estas actividades están estrechamente relacionadas con la gestión de la red LAN.

- **Crear nuevos segmentos de red,** en los últimos años, las universidades han ido creciendo en población estudiantil y con ello se tienen que construir nuevos ambientes académicos o remodelar los existentes.

- **Crear nuevas VLANs,** por la implementación de nuevos proyectos o la ampliación de pabellones o áreas.
- **Reconfiguración de puertos en los conmutadores,** en respuesta a las solicitudes de traslado de personal administrativo o equipos entre áreas y/o pabellones.
- **Implementación de nuevos conmutadores**, cambio de tecnología, ampliación o construcción de nuevas áreas o reemplazo ante averías.
- **Cambio de contraseñas de acceso a los conmutadores,** necesario para la gestión de los conmutadores, ante una falta de gestión centralizada
- **Implementación de nuevas políticas de acceso (ACLs),** necesario ante la creación de nuevos segmentos de red y/o VLANs.
- **Actualización de las políticas de Acceso (ACLs),** en atención de solicitudes de usuarios y/o área para permitir o denegar el acceso a algún servicio o equipo.
- **Actualización del Sistema Operativo de los conmutadores,** ante posibles vulnerabilidades, bug (errores en alguna versión de sistema operativo que genera errores en los conmutadores) o mejoras publicadas por los fabricantes, se tienen que actualizar los sistemas operativos de los conmutadores afectados.

La ejecución de cada una de las actividades antes descritas puede llevar a incurrir en los siguientes problemas:

1 Tiempos muy elevados para el despliegue de configuraciones

El tiempo que le toma al administrador de la red para realizar un cambio en la arquitectura, es bastante elevado. Esto implica que el administrador dedique gran parte de su tiempo a la gestión y configuración de equipamiento, pudiendo utilizar este tiempo a la búsqueda de alternativas de mejora y nuevas soluciones para la red.

Basado en mi experiencia profesional, he elaborado una matriz de evaluación de actividades de gestión de la red por su concurrencia y tiempo de ejecución.

Tabla 4

Matriz de evaluación de actividades de gestión de la red, por concurrencia y tiempo de ejecución

		Concurrencia		
		Bajo	**Medio**	**Alto**
Tiempo de Ejecución	**Bajo**			
	Medio			
	Alto			

Teniendo en consideración la matriz de evaluación presentada en la *Tabla 4*, se evalúan las actividades descritas a continuación:

Tabla 5

Evaluación de las actividades de gestión de la red

Item	Actividades	Concurrencia	Tiempo de Ejecución	Resultado
1	Crear nuevos segmentos de red	**Medio**	**Alto**	
2	Crear nuevas VLANs	**Medio**	**Alto**	
3	Reconfiguración de puertos en los conmutadores	**Alto**	**Medio**	
4	Implementación de nuevos conmutadores	**Medio**	**Alto**	
5	Cambio de contraseñas de acceso a los conmutadores	**Bajo**	**Alto**	
6	Implementación de nuevas políticas de acceso (ACLs)	**Medio**	**Alto**	
7	Actualización de las políticas de Acceso (ACLs)	**Medio**	**Bajo**	
8	Actualización del Sistema Operativo de los conmutadores	**Bajo**	**Alto**	

Tomando como base el resultado de la evaluación realizada en la *Tabla 5*, las actividades que se tomarán en cuenta para un mayor análisis son las que aparecen marcadas de color rojo, en la columna de resultado; es decir, las que son más demandantes en cuanto a concurrencia y tiempo de ejecución.

A continuación, se muestra una tabla con ejemplos del tipo de configuraciones y los tiempos que toma realizarlas

Tabla 6

Medición de Tiempo en la gestión de equipamiento en una red tradicional

Actividad	Detalle	Procedimientos	Tiempo (minutos)	Riesgos	Tiempo Total (minutos)
Crear un nuevo segmento de red	Crear un nuevo segmento de red para una nueva área, servicio o proyecto	Identificar el nuevo segmento de red a ser creado	**45**	Si no se aplica la configuración de la VLAN, de manera correcta, podría dejar sin servicio de reda un área, un pabellón o inclusive a todo el campus universitario.	**150**
		Conectar se al conmutador Core y crear el nuevo segmento de red	**45**		
		Identificar la interfaz física o VLAN a la cual se aplicará el nuevo segmento de red.	**30**		
		Crear la VLAN en caso aún no exista	**30**		
Crear una nueva VLAN	Crear una nueva VLAN y realizar el despliegue la misma en la red.	Conectarse al conmutador Core y crear la VLAN	**30**	Si no se identifica los conmutador y puertos de una manera correcta y luego no se aplica la configuración correctamente, podría quedar sin servicio de red un área, un pabellón o inclusive todo el Campus Universitario.	**120**
		Identificar el(los) conmutador(es) y el(los) puerto(s) para crear y configurar VLAN	**60**		
		Conectarse al(los) conmutador(es) para configurarlos			
		Configurar la VLAN en el(los) puerto(s) del conmutador de manera correcta	**30**		

Actividad	Detalle	Procedimientos	Tiempo (minutos)	Riesgos	Tiempo Total (minutos)
Implementar un nuevo conmutador	Implementación de un nuevo conmutador o reemplazo por uno que falló	Configuración básica (IP Comunidad SNMP, NTP, ETC) del conmutador	**45**	Si no se aplica la configuración al conmutador, de manera correcta, podría no tener servicio el área involucrada, un pabellón o inclusive dejar sin servicio de red a todo el campus universitario.	**225**
		Instalación (rackeo) del conmutador en su ubicación física	**60**		
		Configuración específica del conmutador	**120**		
Reconfiguración de puertos en un conmutador	Actualizar la VLAN en uno o múltiples puertos de un conmutador	Identificar el(los) conmutador(es) y el(los) puerto(s) para crear y configurar VLAN	**60**	Si no se aplica la configuración de la VLAN, de manera correcta, podría dejar sin servicio de reda un área, un pabellón o inclusive a todo el campus universitario.	**90**
		Conectarse al(los) conmutador(es) para configurarlos	**30**		
		Configurar la VLAN en el(los) puerto(s) del conmutador de manera correcta			
Implementar una nueva política de Acceso (ACL)	Implementación de un nuevo conmutador o reemplazo por uno que falló	Analizar el requerimiento (direcciones IP origen, destino y puertos)	**60**	Si no se aplica la configuración al conmutador, de manera correcta, podría no tener servicio el área involucrada, un pabellón o inclusive dejar sin servicio de red a todo el campus universitario.	**155**
		Identificar la red o VLAN a la cual se tiene que aplicar la política.	**30**		
		Pruebas preliminares	**45**		
		Ejecución de la nueva política	**20**		

Los datos mostrados en la *Tabla 6*, están ajustados a la realidad de un campus universitario con una red compleja bajo la arquitectura de red tradicional (Core, Distribución y Acceso) con un administrador y considerando conmutadores de backup en espera.

2 Necesidad de personal altamente capacitado

Adicional al tiempo que toma realizar nuevas configuraciones o actualizaciones sobre la red existente, es importante tener en cuenta, por costos o nuevas características en el funcionamiento de los conmutadores, las redes actuales no cuentan con una única marca de equipos, los mismos pueden ser de las marcas: Cisco, Nokia, Juniper, Huawey, etc,

El contar con conmutadores de distintos fabricantes implica que el personal que gestiona la red tenga conocimiento de todas las marcas y de la infraestructura de red, dificultando aún más la correcta gestión de la red. El desconocimiento o falta de experiencia del personal podría ocasionar que se realice una configuración errónea y dejar sin servicio de red a una parte del campus o inclusive, en un escenario más grave, a todo el campus universitario.

Cada uno de los fabricantes de conmutadores tienen su propia sintaxis de línea de comandos (CLI), los que inclusive ofrecen cursos de especialización con certificación para la gestión y administración de sus dispositivos. En la tabla *Tabla 11*, se muestra un comparativo de la sintaxis de los fabricantes más conocidos a nivel de conmutadores existentes en el mercado.

Figura 23

Comparativo de comandos CLI por fabricantes de conmutadores

Fabricante	CISCO	NOKIA	HUAWEY	JUNIPER
Sistema Operativo	IOS	TiMOS	HVRP	JUNOS
Comando para visualizar parámetros especificos de protocolos y/o sistema	Show	show	display	show
	Running	-	-	do
	Reload	admin reboot now	reboot	request system reboot
	show running-config	admin display-config	display current-configuration	show configuration
	show ntp status	show system ntp	display ntpservice status	show ntp status
	show processes cpu	show system cpu	display cpu-usage	show system processes extensive

Así mismo, para realizar nuevas configuraciones y/o cambios en las configuraciones que impliquen cierta complejidad es necesario se cuente con personal de manera presencial en el

mismo lugar donde están instalados los equipos con el fin de actuar de manera rápida ante una posible incidencia y evitar caídas en el servicio de red.

3 Software de Gestión muy costoso o ausencia de este

Para la gestión de manera centralizada de una red LAN, existen soluciones propietarias. Sin embargo, estas herramientas de gestión son bastante costosas en presupuesto CAPEX y OPEX; además, el contar con una solución propietaria hace que la institución sea dependiente de un mismo fabricante para mantener la compatibilidad de los conmutadores con la herramienta de gestión.

Uniformizar una red, para mantener interoperabilidad de los conmutadores con una herramienta de gestión, implicaría: primero, migrar todos los conmutadores de los diferentes fabricantes a uno solo; y segundo, implementar la herramienta de gestión de acuerdo con el fabricante de conmutadores elegido. Realizar este cambio de infraestructura significa designar un presupuesto bastante elevado para la compra de equipamiento de un mismo fabricante.

Para el escenario propuesto, consideraremos que la red actual está compuesta de conmutadores de los siguientes fabricantes:

- ✓ **El 50% de conmutadores Cisco:** de estos, el 100% son PoE, el 80% son de 48 puertos Gigabit Ethernet y el 20% son de 24 puertos Gigabit Ethernet
- ✓ **El 30% de conmutadores Juniper:** de estos, el 100% son PoE, el 90% son de 48 puertos Gigabit Ethernet y el 10% son de 24 puertos Gigabit Ethernet
- ✓ **El 20% de Conmutadores Dell:** de estos, el 100% son PoE, el 95% son de 48 puertos Gigabit Ethernet y el 5% son de 24 puertos Gigabit Ethernet

En la *Figura 24*, se muestra la distribución de las cantidades de computadores de acuerdo con el fabricante y el número de puertos, esta tabla está elaborada en base al escenario planteado, 100 conmutadores.

Figura 24

Distribución de Conmutadores por fabricante y número de puertos

Fabricante	**CISCO** 50%		**JUNIPER** 30%		**DELL** 20%	
N° Puertos	48 Puertos PoE GE 80%	24 Puertos PoE GE 20%	48 Puertos PoE GE 90%	24 Puertos PoE GE 10%	48 Puertos PoE GE 95%	24 Puertos PoE GE 5%
Total de Conmutadores	**40**	**10**	**27**	**3**	**19**	**1**

En este sentido, para realizar una migración hacia conmutadores de un mismo fabricante, por ejemplo, a Cisco se debe destinar un presupuesto muy elevado. A continuación, se presenta una simulación del costo involucrado considerando que todos los conmutadores Juniper y Dell son del nivel de acceso, lo de distribución y Core son Cisco.

Tabla 7

Cálculo de costos, no se ha considerado el I.G.V., asociado a la migración de 50 conmutadores a la marca Cisco

Item	Modelo	N° Puertos	Cantidad	Costo Unitario	Costo Total
1	WS-C2960X-48FPS-L	48	46	$7,480.00	$344,080.00
2	WS-C2960X-24PS-L	24	4	$3,466.00	$13,864.00
		TOTAL			$357,944.00

A modo de ilustración, por ejemplo, para una universidad que migró toda su infraestructura de conmutadores de la marca Cisco y lograr gestionarlos de manera centralizada, tendría que adquirir el software de gestión, Cisco Prime Infrastructure. Sin embargo, para implementar una solución de este tipo y mantenerla en el tiempo, hace que la institución disponga de un presupuesto considerable. Tal como se muestra en el *Tabla 8*. También se detallan los costos de licenciamiento y soporte anual. El costo por la implementación de la herramienta no lo está siendo considerado en el presente análisis.

Es importante recalcar que el costo de licenciamiento mostrado en la siguiente tabla únicamente corresponde a 120 licencias; es decir, solo se podrán gestionar 120 conmutadores como máximo.

Tabla 8

Costo por licenciamiento y soporte para el Cisco Prime Infrastructure

Software de Gestión	Tipo de licenciamiento	N° de Licencias requeridas	Costo por licenciamiento	Precio por el Soporte Anual	Precio Total
Cisco Prime Infrastructure	Este software se licencia por equipo o dispositivo gestionado	120	$12,600.00	$12,409.72	$25,009.72

Nota. Los precios que se detallan no consideran el I.G.V.

Este tipo de softwares o herramientas de gestión no siempre se terminan alineando a las necesidades de las organizaciones debido a que no permiten un desarrollo personalizado.
Finalmente, sin un software de gestión no es posible tener un monitoreo centralizado de todos los conmutadores lo que significa que no se tendrá visibilidad de la salud o disponibilidad de cada uno de ellos. Esto también implica que, si no hay un monitoreo de los equipos, tampoco se podrá tener alertas o avisos para una rápida reacción de los administradores de la red ante la posible falla de uno o varios conmutadores.

3.3. Requerimientos.

Los requerimientos que se debe tener en cuenta para para el diseño de la red SDN, son los que a continuación se describen:

- ✓ **R1: Se debe poder implementar políticas de acceso (ACLs)**

Se necesita que la red SDN tenga la posibilidad de poder implementar listas de control de acceso (ACLs) con el fin de poder permitir o denegar cierto tráfico en la red LAN

- ✓ **R2: Despliegue de configuraciones a un conmutador**

Es necesario que la red SDN tenga la capacidad de poder hacer despliegue de configuraciones y la posibilidad de elegir cualquier conmutador de la red de manera independiente desde un punto centralizado.

- ✓ **R3: Despliegue de configuraciones a un grupo de conmutadores**

La red SDN, debe tener la capacidad de poder hacer despliegue de configuraciones y la posibilidad de elegir un grupo de conmutadores de la red desde un punto centralizado.

- ✓ **R4: Despliegue de configuraciones a todos los conmutadores**

La red SDN, debe tener la capacidad de poder hacer despliegue de configuraciones a todos los conmutadores de la red desde un punto centralizado

- ✓ **R5: Despliegue configuraciones de manera automatizada**

El sistema de gestión de la red debe tener la capacidad de enviar configuraciones a los conmutadores de manera automatizada, es decir, el administrador de la red deber poder programar hora y fecha de acuerdo a lo que mejor considere.

- ✓ **R6: Monitorear la red de manera centralizada**

Es necesario que el sistema centralizado de la red SDN tenga la capacidad de monitorear la salud de los conmutadores (CPU, Memoria RAM y conectividad).

✓ **R7: Generar alertas ante la falla de un conmutador**

El sistema de monitoreo debe tener la capacidad de generar alertas ante la falla de uno o varios conmutadores.

✓ **R8: Respaldo de configuraciones de los equipos**

El sistema de gestión de la red SDN, debe tener la capacidad de realizar un respaldo de las configuraciones de los controladores del HA

✓ **R9: Soporte para implementar alta disponibilidad a nivel de capa de control**

El diseño debe tener una alta disponibilidad a nivel de capa de control, es decir ante la falla de un controlador debe existir otro que cumpla su función.

✓ **R10: Soporte para balanceo de carga a nivel de controladores**

Los controladores implementados en alta disponibilidad deben soportar la funcionalidad de balanceo de carga.

✓ **R11: Almacenamiento y despliegue de plantillas de configuración.**

Se debe tener la capacidad de almacenar plantillas de configuración para ser utilizadas ante la instalación de un nuevo conmutador.

✓ **R12: Interoperabilidad entre distintos fabricantes de conmutadores**

La red SDN debe tener la capacidad de permitir la interoperabilidad entre equipos conmutadores de distintos fabricantes

✓ **R13: Escalabilidad**

La red SDN debe tener la capacidad de integrar nuevos conmutadores ante el crecimiento de la red sin la necesidad de nuevos licenciamientos.

✓ **R14: Baja demanda de recursos de hardware para la gestión centralizada**

La implementación de la gestión centralizada no debe ser altamente demandante de recursos de hardware.

3.4. Objetivos Específicos vs Requerimientos

A continuación, en la *Tabla 9*, se detallan todos los requerimientos y la vinculación con el objetivo específico que corresponde.

Tabla 9

Relacionamiento de los requerimientos con los objetivos específicos

OBJETIVO	REQUERMIENTO	DESCRIPCIÓN
OE1: Gestionar todos los conmutadores que conforman la red desde un punto central, con la finalidad de que todos sean monitoreados y gestionados de manera centralizada.	**R1:** Se debe poder implementar políticas de acceso (ACLs)	La red SDN debe tener la posibilidad de poder implementar listas de control de acceso (ACLs)
	R6: Monitoreo la red de manera centralizada	El sistema centralizado de la red SDN debe tener la capacidad de monitorear la salud de los conmutadores (CPU, Memoria RAM y conectividad).
	R7: Enviar alertas ante fallas de un conmutador	El sistema de monitoreo debe tener la capacidad de generar alertas ante la falla de uno o varios conmutadores.
	R8: Respaldo de configuraciones de los equipos	El sistema de gestión de la red SDN, debe tener la capacidad de realizar un respaldo de las configuraciones de los controladores del HA
OE2: Mantener un arreglo de controladores SDN en alta disponibilidad y permita el balanceo de carga en la gestión de los conmutadores, de tal manera que estos no pierdan conectividad con el controlador	**R9:** Soporte para implementar alta disponibilidad a nivel de capa de control	El diseño debe tener una alta disponibilidad a nivel de capa de control, es decir ante la falla de un controlador debe existir otro que cumpla su función
	R10: Soporte para balanceo de carga a nivel de controladores	Los controladores implementados en alta disponibilidad deben soportar la funcionalidad de balanceo de carga.
OE3: Disminuir los tiempos de despliegue de configuraciones y actualizaciones de la red.	**R2:** Despliegue de configuraciones a un conmutador	La red debe tener la capacidad de poder hacer despliegue de configuraciones y la posibilidad de elegir cualquier conmutador de la red de manera independiente desde un punto centralizado.
	R3: Despliegue de configuraciones a un grupo de conmutadores	La red debe tener la capacidad de poder hacer despliegue de configuraciones y la posibilidad de elegir un grupo de conmutadores de la red desde un punto centralizado.

	R4: Despliegue de configuraciones a todos los conmutadores	La red, debe tener la capacidad de poder hacer despliegue de configuraciones a todos los conmutadores de la red desde un punto centralizado
	R5: Despliegue configuraciones de manera automatizada	Debe tener la capacidad de enviar configuraciones a los conmutadores de manera automatizada.
	R11: Almacenamiento y despliegue de plantillas de configuración.	Se debe tener la capacidad de almacenar plantillas de configuración para ser utilizadas ante la instalación de un nuevo conmutador.
OE4: Proponer una optimización en los costos de inversión para la implementación de red LAN con su gestión centralizada.	**R12:** Interoperabilidad entre distintos fabricantes de conmutadores	La red SDN debe tener la capacidad de permitir la interoperabilidad entre equipos conmutadores de distintos fabricantes
	R13: Escalabilidad	La red SDN debe tener la capacidad de integrar nuevos conmutadores ante el crecimiento de la red
	R14: Baja demanda de recursos de hardware para la gestión centralizada	La implementación de la gestión centralizada no debe ser altamente demandante de recursos de hardware.

4. CAPÍTULO 4: DISEÑO DE LA SOLUCION

Siguiendo las recomendaciones y mejores prácticas establecidas por la ONF se ha realizado la selección del modelo de despliegue SDN, controlador SDN, y versión del protocolo OpenFlow.

4.1. Selección del modelo de despliegue de Red SDN

La adopción de una nueva arquitectura de red es un tema muy complejo, generalmente en una migración hacia una nueva arquitectura, los administradores de red son reacios al cambio; estas reacciones se deben principalmente porque tienen que adquirir nuevas habilidades relacionadas a la nueva arquitectura de red. En este sentido y en función de lo descrito, en el marco teórico, de cada uno de los modelos de despliegue: modelo SDN basado en dispositivos, modelo SDN overlay y modelo SDN híbrido; el modelo más adecuado para el diseño de la red SDN del presente proyecto, es el **modelo SDN basado en dispositivos**.

La elección del modelo de red está basada principalmente a las siguientes razones:

a) El modelo de red SDN basado en dispositivos es el modelo más fácil de implementar debido a que la nueva red SDN no tendrá la necesidad de coexistir ni tener que interoperar con la antigua red convencional.
b) El beneficio de este modelo consiste en la posibilidad de cambiar conmutadores tradicionales por conmutadores OpenFlow, este modelo puede brindar la facilidad de implementar una estrategia por fases. Las fases pueden consistir en migrar la red en su totalidad de un pabellón o pisos de un pabellón que tengan una red LAN independiente, es decir, no compartan conmutadores con otros pisos. De esta manera se puede ir migrando la red de manera gradual hasta llegar a tener todo el campus con equipos 100% OpenFlow.

4.2. Selección del Controlador SDN

4.2.1. Controlador

El controlador ONOS, ha sido elegido debido a las bondades y características que ofrece; entre las más resaltantes para este proyecto, se detallan a continuación:

- ✓ Capacidades de brindar alta disponibilidad a nivel de controlador
- ✓ Facilidad de implementación
- ✓ Poco uso de recursos de hardware
- ✓ Interfaces (WEB y CLI) muy amigables e intuitivas para el usuario
- ✓ Alta disponibilidad con balanceo de carga

4.2.2. Versión del Controlador

Cómo bien se mencionó en el marco teórico, el controlador ONOS cuenta con múltiples versiones. Sin embargo, en este diseño se está considerando la versión 1.13 (Nightingale), la revisión 10. Esta versión 1.13.10 fue publicada el 20 de febrero del 2020. (Ayaka Koshibe, 2022)

Esta versión del controlador ONOS, utiliza la versión 8 de Java

4.2.1. Alta Disponibilidad de Controlador

La alta disponibilidad a nivel del controlador ONOS seleccionado se logra utilizando 3 nodos, en nuestro caso consideraremos máquinas virtuales para montar cada uno de los nodos, es decir se consideran tres controladores los mismos que deben estar alojados en distintas máquinas virtuales. Estos tres controladores formarán un cluster.

Para tener la alta disponibilidad en la red de este proyecto, todos los conmutadores deben estar conectados de manera lógica a cada uno de los controladores, es decir, en la configuración del conmutador respecto del controlador, se debe considerar las direcciones IP y puertos de cada uno de los tres controladores que conforman el cluster.

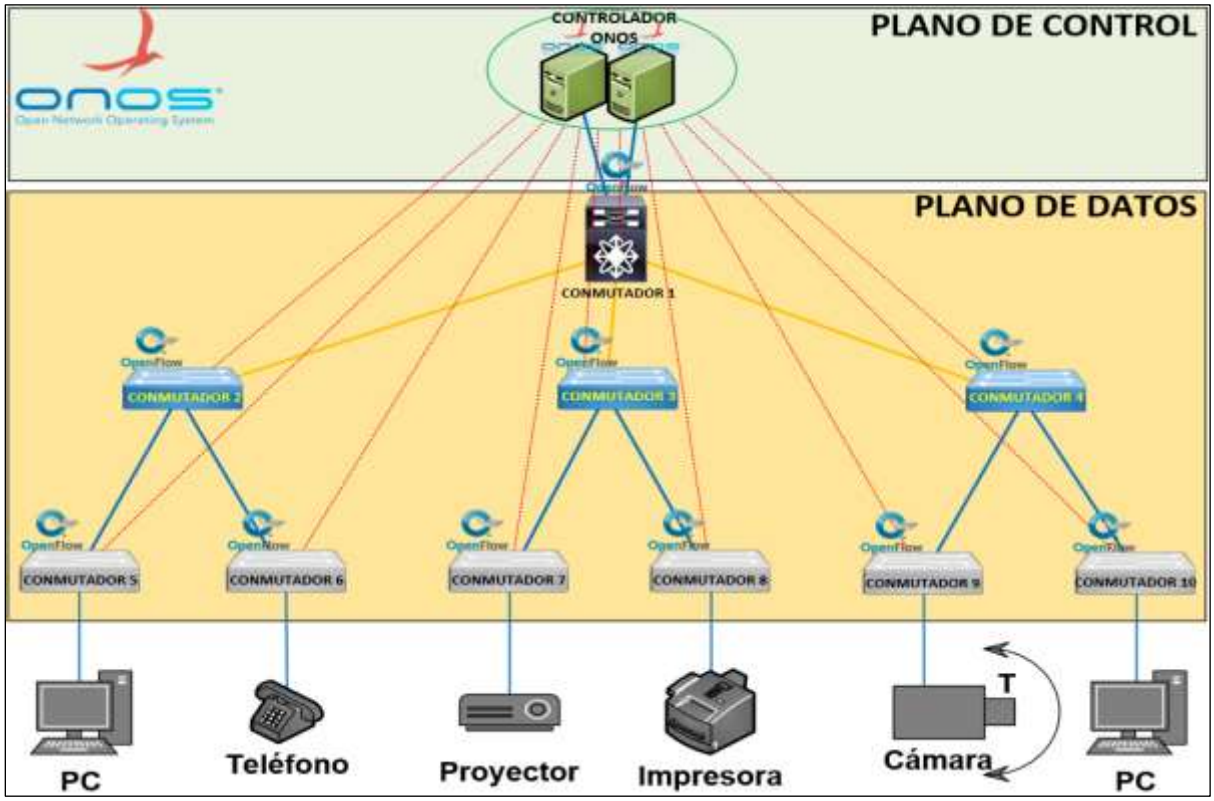
Así mismo, para reducir la carga que realizarán cada uno de los controladores, haremos uso de la funcionalidad de balanceo de carga que ofrece ONOS. De esta manera, los 3 nodos del cluster compartirán la carga de la red SDN.

4.3. Topología de Red SDN

En la figura 25 se muestra un extracto del diagrama general de red SDN contemplado para el presente proyecto, en este diagrama se muestra claramente el plano de datos y el plano de control, así como también se puede apreciar la forma de conexión física y lógica de los conmutadores con el controlador.

Figura 25

Extracto del Diagrama general de red SDN



a) Plano de control

En el plano de control se encuentra ubicado el controlador SDN, tal como se mencionó en el punto 4.2, el controlador elegido para este proyecto es ONOS. De acuerdo con la figura 25, el diseño contempla a un controlador en alta disponibilidad para evitar que los conmutadores pierdan gestión en algún momento.

b) Plano de datos

Aquí se encuentra ubicados todos los conmutadores, los que son controlados por el controlador ONOS y que a su vez brindan conectividad a los dispositivos de red finales, es decir, computadoras, impresoras, proyectores, teléfono, cámaras de video vigilancia, etc.

c) Conexión Física

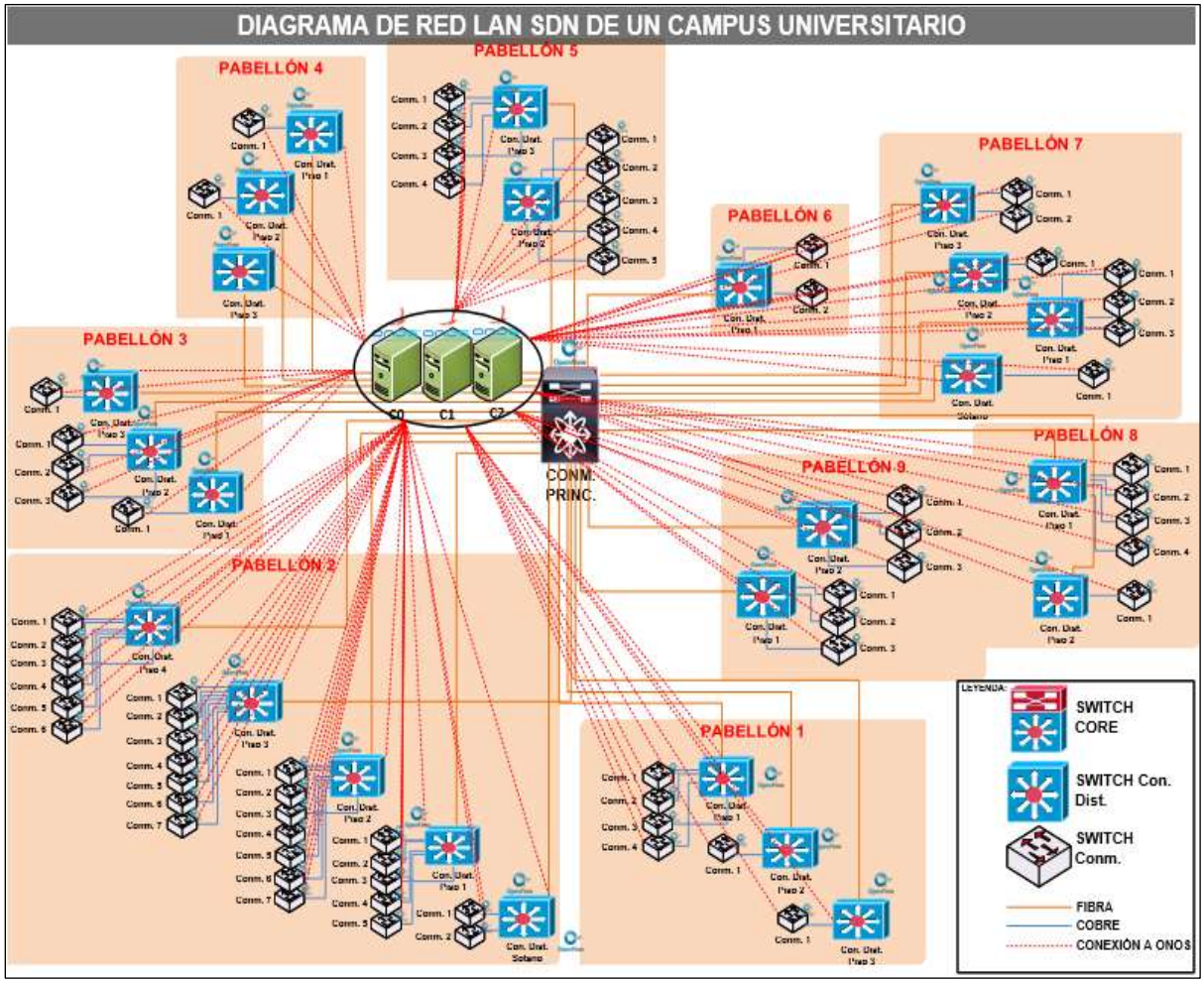
De manera física, dado que los conmutadores en un campus universitario no se encuentran en una sola ubicación; sino por lo contrario podrían ubicarse en distintos edificios y en diferentes pisos, es necesario se tenga una topología del tipo árbol tal como se muestra en el diagrama de la figura 26. De esta manera se cuenta con un conmutador principal, el cual podría ubicarse en el centro de datos de la universidad junto con el controlador SDN.

Al conmutador principal se conectan de manera física el controlador SDN y los conmutadores concentradores de un edificio o un piso, que para mejor entendimiento podemos llamarlos conmutadores de distribución.

A los conmutadores de distribución se conectan los conmutadores finales, que, si queremos compararlos con una red tradicional, mostrada en la figura 21, podemos decir que serían los conmutadores de acceso los cuales brindan conectividad a los dispositivos de red finales.

Figura 26

Diagrama general de red SDN



d) Conexión Lógica

A nivel lógico, todos los conmutadores son directamente controlados y gestionados por el controlador SDN, para el envío del tráfico de control entre el controlador y los conmutadores se usará el canal de control fuera de banda (out-of-band control) haciendo uso de una vlan independiente para este fin. Tal como se muestra en la figura 25 y 26, todos los conmutadores

se encuentran conectados al controlador, esta conexión está representada por las líneas punteadas de color rojo.

4.4. Recursos

En la tabla 10 se detallan los requisitos y mínimos y recomendados necesarios respecto a Hardware y Software para la instalación del controlador ONOS.

Cómo nuestro diseño contempla tres nodos de controlador para formar el cluster, la información que se muestra en la tabla 10 representa los recursos de hardware necesarios por cada uno de los controladores.

Tabla 10

Requisitos de Hardware para el controlador ONOS

CONTROLADOR	REQUERIMIENTOS MÍNIMOS DEL SISTEMA	REQUERIMIENTOS RECOMENDAODOS DEL SISTEMA	SISTEMA OPERATIVOS RECOMENDADOS
ONOS-1	✓ CPU: 2 Cores ✓ RAM: 2 GB ✓ DISCO: 10 GB	✓ CPU: 4 Cores ✓ RAM: 4 GB ✓ DISCO: 100 GB	✓ Ubuntu 16.X o superior ✓ Java Virtual Machine Ver. 8
ONOS-2	✓ CPU: 2 Cores ✓ RAM: 2 GB DISCO: 10 GB	✓ CPU: 4 Cores ✓ RAM: 4 GB DISCO: 100 GB	✓ Ubuntu 16.X o superior Java Virtual Machine Ver. 8
ONOS-3	✓ CPU: 2 Cores ✓ RAM: 2 GB DISCO: 10 GB	✓ CPU: 4 Cores ✓ RAM: 4 GB DISCO: 100 GB	✓ Ubuntu 16.X o superior Java Virtual Machine Ver. 8

Nota. Adaptado de "Descarga de versiones de ONOS", por Ayaka Koshibe, 2022 (https://wiki.onosproject.org/display/ONOS/Downloads).

Con el propósito de no depender únicamente de un solo controlador, en el proyecto se considerará tres controladores para lograr alta disponibilidad; de esta manera, ante la caída o falla de uno, los dos controladores restantes asumirán la carga de los conmutadores.

4.5. Desarrollo del diseño

En este punto se realiza el desarrollo del diseño de red SDN que se plantea en este proyecto.

4.5.1. Direccionamiento IP de la red

Para el direccionamiento de direcciones IP se utilizarán la versión 4 de IP y se describe en las siguientes tablas direccionamiento de la red utilizado será el siguiente

a) Segmento de red por ámbitos

En este apartado se describe los segmentos de red por cada uno de los ámbitos que tendrá de manera independiente una red (VLAN).

En la siguiente tabla se describe el ámbito, el segmento de red asignado con su respectiva máscara de red, la cantidad de host (equipos) que se podrán tener y el número de VLAN correspondiente.

Tabla 11

Segmentación de red del escenario trabajado

ÁMBITO	SEGMENTO DE RED	MASCARA DE RED	CANTIDAD DE HOST	VLAN
Servidores Administrativos	10.20.20.0	255.255.255.0	254	20
Servidores Académicos	10.20.22.0	255.255.255.0	254	22
WiFi Administrativo	10.20.30.0	255.255.255.0	254	30
	10.20.31.0	255.255.255.0	254	31
WiFi Alumnos	10.20.32.0	255.255.254.0	510	32
	10.20.34.0	255.255.254.0	510	34
	10.20.36.0	255.255.254.0	510	36
	10.20.38.0	255.255.254.0	510	38
	10.20.40.0	255.255.254.0	510	40
	10.20.42.0	255.255.254.0	510	42
	10.20.44.0	255.255.254.0	510	44
	10.20.46.0	255.255.254.0	510	46
	10.20.48.0	255.255.254.0	510	48
	10.20.50.0	255.255.254.0	510	50
	10.20.52.0	255.255.254.0	510	52
	10.20.54.0	255.255.254.0	510	54
	10.20.56.0	255.255.254.0	510	56
	10.20.58.0	255.255.254.0	510	58
WiFi Docentes	10.20.66.0	255.255.254.0	510	66
	10.20.68.0	255.255.254.0	510	68
WiFi Invitados	10.20.70.0	255.255.255.0	254	70
	10.20.71.0	255.255.255.0	254	71
Laboratorios Windows	10.20.72.0	255.255.255.0	254	72
	10.20.73.0	255.255.255.0	254	73
	10.20.74.0	255.255.255.0	254	74
Laboratorios MAC	10.20.77.0	255.255.255.0	254	77
	10.20.78.0	255.255.255.0	254	78

Impresoras	10.20.80.0	255.255.255.0	254	80
Proyectores	10.20.82.0	255.255.255.0	254	82
Aulas	10.20.84.0	255.255.255.0	254	84
Gestión de Conmutadores	10.20.200.0	255.255.255.0	254	200
Gestión de Antenas	10.20.202.0	255.255.255.0	254	202
WiFi	10.20.203.0	255.255.255.0	254	203
Gestión de Servidores	10.20.205.0	255.255.255.0	254	205

b) Descripción de los segmentos de red.

En esta tabla se describen la utilidad que se le dará a cada uno de los segmentos de red y los equipos que podrán estar dentro de cada uno.

Tabla 12

Descripción de los segmentos de red

ÁMBITO	VLAN	Descripción
Servidores Administrativos	20	En esta VLAN estarán todos los servidores correspondientes a los servicios administrativos (Directorio Activo, DNS, DHCP, Correo, etc.
Servidores Académicos	22	En esta VLAN se encontrarán todos los servidores asignados para desarrollos y pruebas de las áreas académicas
WiFi Administrativo	30	Dedicada para la asignación de IPs a los usuarios administrativos
	31	Dedicada para la asignación de IPs a los usuarios administrativos
WiFi Alumnos	32	Dedicada para la asignación de IPs a los estudiantes
	34	Dedicada para la asignación de IPs a los estudiantes
	36	Dedicada para la asignación de IPs a los estudiantes
	38	Dedicada para la asignación de IPs a los estudiantes
	40	Dedicada para la asignación de IPs a los estudiantes
	42	Dedicada para la asignación de IPs a los estudiantes
	44	Dedicada para la asignación de IPs a los estudiantes
	46	Dedicada para la asignación de IPs a los estudiantes
	48	Dedicada para la asignación de IPs a los estudiantes
	50	Dedicada para la asignación de IPs a los estudiantes

WiFi Docentes	52	Dedicada para la asignación de IPs a los estudiantes
	54	Dedicada para la asignación de IPs a los estudiantes
	56	Dedicada para la asignación de IPs a los estudiantes
	58	Dedicada para la asignación de IPs a los estudiantes
	66	Dedicada para la asignación de IPs a los docentes
	68	Dedicada para la asignación de IPs a los docentes
WiFi Invitados	70	Dedicada para la asignación de IPs a personal externa a la Universidad (visitas, proveedores, etc)
Laboratorios Windows	71	Dedicada para la asignación de IPs a las computadoras de los laboratorios con Sistema Operativo Windows
	72	Dedicada para la asignación de IPs a las computadoras de los laboratorios con Sistema Operativo Windows
	73	Dedicada para la asignación de IPs a las computadoras de los laboratorios con Sistema Operativo Windows
	74	Dedicada para la asignación de IPs a las computadoras de los laboratorios con Sistema Operativo Windows
Laboratorios MAC	77	Dedicada para la asignación de IPs a las computadoras de los laboratorios con Sistema Operativo Apple
	78	Dedicada para la asignación de IPs a las computadoras de los laboratorios con Sistema Operativo Apple
Impresoras	80	Dedicada para la asignación de IPs a las Impresoras
Proyectores	82	Dedicada para la asignación de IPs a los proyectores
Aulas	84	Dedicada para la asignación de IPs a las computadoras de las aulas
Gestión de Conmutadores	200	Dedicada para la asignación de IPs a los conmutadores de todo el campus universitario
Gestión de Antenas WiFi	202	Dedicada para la asignación de IPs a las antenas WiFi (puntos de acceso) de todo el campus universitario
	203	Dedicada para la asignación de IPs a las antenas WiFi (puntos de acceso) de todo el campus universitario
Gestión de Servidores	205	Dedicada para la asignación de IPs a las a los puertos de gestión de los servidores

c) Direccionamiento IP por equipos

En la siguiente tabla se detalla el direccionamiento IP de cada uno de los equipos (conmutadores, controladores ONOS, servidores DHCP, etc.) que forman parte del diseño de la red.

Tabla 13

Direccionamiento IP por equipos

FUNCIÓN	HOSTNAME	DIRECCIÓN IP	GATEWAY	VLAN
Centro de Datos				
Controlador	ONOS-1	10.20.20.11	10.20.20.1	20
Controlador	ONOS-2	10.20.20.12	10.20.20.1	20
Controlador	ONOS-3	10.20.20.13	10.20.20.1	20
Servidor DHCP	DHCP-SERVER	10.20.20.5	10.20.20.1	20
Servidor DNS	DNS-SERVER	10.20.20.6	10.20.20.1	20
Conmutador Principal	CORE	10.20.200.10	10.20.200.1	200
Conmutador Acceso	ACCE-CD-1	10.20.200.11	10.20.200.1	200
Pabellón 1 / Piso 1				
Conmutador Distribución	DIST-PAB1-P1	10.20.200.20	10.20.200.1	200
Conmutador Acceso	ACCE-PAB1-P1-1	10.20.200.21	10.20.200.1	200
Conmutador Acceso	ACCE-PAB1-P1-2	10.20.200.22	10.20.200.1	200
Conmutador Acceso	ACCE-PAB1-P1-3	10.20.200.23	10.20.200.1	200
Conmutador Acceso	ACCE-PAB1-P1-4	10.20.200.24	10.20.200.1	200
Pabellón 1 / Piso 2				
Conmutador Distribución	DIST-PAB1-P2	10.20.200.27	10.20.200.1	200
Conmutador Acceso	ACCE-PAB1-P2-1	10.20.200.28	10.20.200.1	200
Pabellón 1 / Piso 3				
Conmutador Distribución	DIST-PAB1-P3	10.20.200.30	10.20.200.1	200
Conmutador Acceso	ACCE-PAB1-P3-1	10.20.200.31	10.20.200.1	200
Pabellón 1 / Piso 3				
Conmutador Distribución	DIST-PAB2-S1	10.20.200.34	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-S1-1	10.20.200.35	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-S1-2	10.20.200.36	10.20.200.1	200
Pabellón 2 / Piso 1				
Conmutador Distribución	DIST-PAB2-P1	10.20.200.39	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P1-1	10.20.200.40	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P1-2	10.20.200.41	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P1-3	10.20.200.42	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P1-4	10.20.200.43	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P1-5	10.20.200.44	10.20.200.1	200
Pabellón 2 / Piso 2				
Conmutador Distribución	DIST-PAB2-P2	10.20.200.47	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P2-1	10.20.200.48	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P2-2	10.20.200.49	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P2-3	10.20.200.50	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P2-4	10.20.200.51	10.20.200.1	200

Conmutador Acceso	ACCE-PAB2-P2-5	10.20.200.52	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P2-6	10.20.200.53	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P2-7	10.20.200.54	10.20.200.1	200
	Pabellón 2 / Piso 3			
Conmutador Distribución	DIST-PAB2-P3	10.20.200.57	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P3-1	10.20.200.58	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P3-2	10.20.200.59	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P3-3	10.20.200.60	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P3-4	10.20.200.61	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P3-5	10.20.200.62	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P3-6	10.20.200.63	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P3-7	10.20.200.64	10.20.200.1	200
	Pabellón 2 / Piso 4			
Conmutador Distribución	DIST-PAB2-P4	10.20.200.67	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P4-1	10.20.200.68	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P4-2	10.20.200.69	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P4-3	10.20.200.70	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P4-4	10.20.200.71	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P4-5	10.20.200.72	10.20.200.1	200
Conmutador Acceso	ACCE-PAB2-P4-6	10.20.200.73	10.20.200.1	200
	Pabellón 3 / Piso 1			
Conmutador Distribución	DIST-PAB3-P1	10.20.200.76	10.20.200.1	200
Conmutador Acceso	ACCE-PAB3-P1-1	10.20.200.77	10.20.200.1	200
	Pabellón 3 / Piso 2			
Conmutador Distribución	DIST-PAB3-P2	10.20.200.80	10.20.200.1	200
Conmutador Acceso	ACCE-PAB3-P2-1	10.20.200.81	10.20.200.1	200
Conmutador Acceso	ACCE-PAB3-P2-2	10.20.200.82	10.20.200.1	200
Conmutador Acceso	ACCE-PAB3-P2-3	10.20.200.83	10.20.200.1	200
	Pabellón 3 / Piso 3			
Conmutador Distribución	DIST-PAB3-P3	10.20.200.86	10.20.200.1	200
Conmutador Acceso	ACCE-PAB3-P3-1	10.20.200.87	10.20.200.1	200
	Pabellón 4 / Piso 1			
Conmutador Distribución	DIST-PAB4-P1	10.20.200.90	10.20.200.1	200
Conmutador Acceso	ACCE-PAB4-P1-1	10.20.200.91	10.20.200.1	200
	Pabellón 4 / Piso 2			
Conmutador Distribución	DIST-PAB4-P2	10.20.200.94	10.20.200.1	200
Conmutador Acceso	ACCE-PAB4-P2-1	10.20.200.95	10.20.200.1	200
	Pabellón 4 / Piso 3			
Conmutador Distribución	DIST-PAB4-P3	10.20.200.98	10.20.200.1	200
	Pabellón 5 / Piso 2			
Conmutador Distribución	DIST-PAB4-P2	10.20.200.101	10.20.200.1	200
Conmutador Acceso	ACCE-PAB5-P2-1	10.20.200.102	10.20.200.1	200
Conmutador Acceso	ACCE-PAB5-P2-2	10.20.200.103	10.20.200.1	200
Conmutador Acceso	ACCE-PAB5-P2-3	10.20.200.104	10.20.200.1	200
Conmutador Acceso	ACCE-PAB5-P2-4	10.20.200.105	10.20.200.1	200
Conmutador Acceso	ACCE-PAB5-P2-5	10.20.200.106	10.20.200.1	200
	Pabellón 5 / Piso 3			
Conmutador Distribución	DIST-PAB5-P3	10.20.200.109	10.20.200.1	200
Conmutador Acceso	ACCE-PAB5-P3-1	10.20.200.110	10.20.200.1	200
Conmutador Acceso	ACCE-PAB5-P3-2	10.20.200.111	10.20.200.1	200

Conmutador Acceso	ACCE-PAB5-P3-3	10.20.200.112	10.20.200.1	200
Conmutador Acceso	ACCE-PAB5-P3-4	10.20.200.113	10.20.200.1	200
	Pabellón 6 / Piso 1			
Conmutador Distribución	DIST-PAB6-P1	10.20.200.116	10.20.200.1	200
Conmutador Acceso	ACCE-PAB6-P1-1	10.20.200.117	10.20.200.1	200
Conmutador Acceso	ACCE-PAB6-P1-2	10.20.200.118	10.20.200.1	200
	Pabellón 7 / Sótano 1			
Conmutador Distribución	DIST-PAB7-S1	10.20.200.121	10.20.200.1	200
Conmutador Acceso	ACCE-PAB7-S1-1	10.20.200.122	10.20.200.1	200
	Pabellón 7 / Piso 1			
Conmutador Distribución	DIST-PAB7-P1	10.20.200.125	10.20.200.1	200
Conmutador Acceso	ACCE-PAB7-P1-1	10.20.200.126	10.20.200.1	200
Conmutador Acceso	ACCE-PAB7-P1-2	10.20.200.127	10.20.200.1	200
Conmutador Acceso	ACCE-PAB7-P1-3	10.20.200.128	10.20.200.1	200
	Pabellón 7 / Piso 2			
Conmutador Distribución	DIST-PAB7-P2	10.20.200.131	10.20.200.1	200
Conmutador Acceso	ACCE-PAB7-P2-1	10.20.200.132	10.20.200.1	200
	Pabellón 7 / Piso 3			
Conmutador Distribución	DIST-PAB7-P3	10.20.200.135	10.20.200.1	200
Conmutador Acceso	ACCE-PAB7-P3-1	10.20.200.136	10.20.200.1	200
Conmutador Acceso	ACCE-PAB7-P3-2	10.20.200.137	10.20.200.1	200
	Pabellón 8 / Piso 1			
Conmutador Distribución	DIST-PAB8-P1	10.20.200.140	10.20.200.1	200
Conmutador Acceso	ACCE-PAB8-P1-1	10.20.200.141	10.20.200.1	200
Conmutador Acceso	ACCE-PAB8-P1-2	10.20.200.142	10.20.200.1	200
Conmutador Acceso	ACCE-PAB8-P1-3	10.20.200.143	10.20.200.1	200
Conmutador Acceso	ACCE-PAB8-P1-4	10.20.200.144	10.20.200.1	200
	Pabellón 8 / Piso 2			
Conmutador Distribución	DIST-PAB8-P2	10.20.200.147	10.20.200.1	200
Conmutador Acceso	ACCE-PAB8-P2-1	10.20.200.148	10.20.200.1	200
	Pabellón 9 / Piso 1			
Conmutador Distribución	DIST-PAB9-P1	10.20.200.151	10.20.200.1	200
Conmutador Acceso	ACCE-PAB9-P1-1	10.20.200.152	10.20.200.1	200
Conmutador Acceso	ACCE-PAB9-P1-2	10.20.200.153	10.20.200.1	200
Conmutador Acceso	ACCE-PAB9-P1-3	10.20.200.154	10.20.200.1	200
	Pabellón 9 / Piso 2			
Conmutador Distribución	DIST-PAB9-P2	10.20.200.157	10.20.200.1	200
Conmutador Acceso	ACCE-PAB9-P2-1	10.20.200.158	10.20.200.1	200
Conmutador Acceso	ACCE-PAB9-P2-2	10.20.200.159	10.20.200.1	200
Conmutador Acceso	ACCE-PAB9-P2-3	10.20.200.160	10.20.200.1	200

4.5.2. Instalación del Controlador ONOS

En este apartado se desarrolla todo lo necesario para que el controlador ONOS sea implementado.

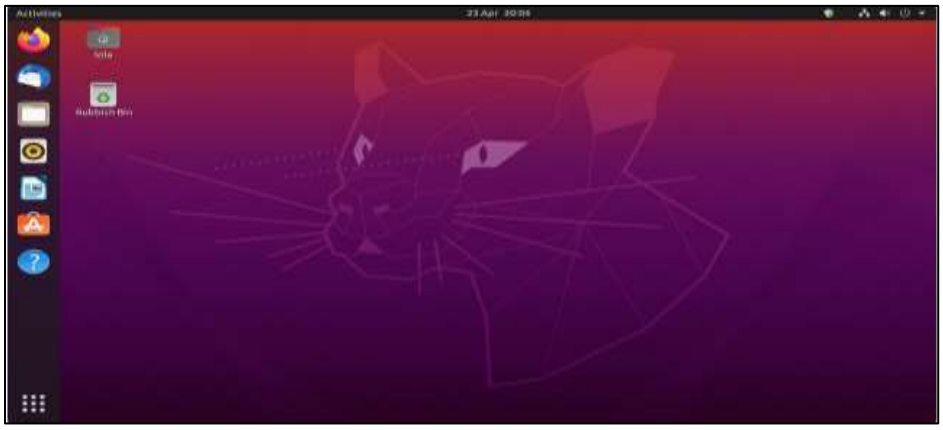
a) Sistema Operativo

El controlador SDN ONOS necesita como base, para su instalación, un sistema Operativo Linux.

El Sistema Operativo que utilizaremos en este proyecto será Linux Ubuntu, la versión 20.04. Este Sistema Operativo será el que contenga el controlador, dentro del cual se activarán todas las características necesarias para el funcionamiento de la red.

Figura 27

Linux Ubuntu 20.04



b) Instalación de Java

Para que el controlador ONOS versión 1.13.10 funcione, es necesario tener instalado el Java versión 8. Sin embargo, Ubuntu 20.04 no trae por defecto la versión 8 de java; en tal sentido, lo que se realizará será la instalación de java para tener compatibilidad con la versión del controlador.

Como primer paso ejecutamos el siguiente comando:

~# sudo apt install -y openjdk-8-jre

Para el paquete de desarrollo JDK de OpenJDK 8, instalaremos el empaquetado openjdk-8-jdk:

~# sudo apt install -y openjdk-8-jdk

Como pueden existir múltiples versiones de java instaladas, ejecutamos el siguiente comando para elegir la versión predeterminada.

~# sudo update-alternatives --config java

Figura 28

Elección de versión predeterminada de Java

Finalmente verificamos la versión que está de manera predeterminada

Figura 29

Validación de la versión de Java

c) Instalación del ONOS

Como primer paso se debe descargar el paquete de instalación con el siguiente comando:

~# wget https://repo1.maven.org/maven2/org/onosproject/onos-releases/1.13.10/onos-1.13.10.tar.gz

Crear la carpeta opt en la raíz y acceder a ella.

~# sudo mkdir -p /opt && cd /opt

Luego de finalizada la descarga, extraer el archivo tar:

~# tar -xvf onos-1.13.10.tar.gz

El archivo se descomprime en una carpeta de nombre onos-1.10.13, se debe cambiar de nombre a la carpeta. El nombre final de la carpeta será onos

~# sudo mv onos-1.10.13 onos

El siguiente paso consiste en hacer que el servicio onos se inicie de manera automática al reiniciar el sistema operativo.

~# sudo cp /opt/onos/init/onos.initd /etc/init.d/onos

~# sudo cp /opt/onos/init/onos.service /etc/systemd/system/

~# sudo systemctl daemon-reload

~# sudo systemctl enable onos

Luego iniciamos el servicio onos

~# sudo systemctl start onos

Para ingresar a la interfaz CLI del controlador podemos hacerlo mediante la siguiente sitaxis de comandos, el usuario por defecto es **onos** y la contraseña es **rocks**

~# ssh -p 8101 onos@ip_onos

Figura 30

Interfaz CLI del controlador ONOS

Para ingresar a la interfaz web de onos se debe ingresar mediante un navegador web haciendo uso de la contraseña y contraseña por defecto ***onos/rocks***.
http://ip_onos:8181/onos/ui/index.html

Figura 31

Interfaz WEB del controlador ONOS

4.5.3. Instalar de aplicaciones

El controlador ONOS trabaja mediante aplicaciones para las distintas funcionalidades que ofrece. Por ejemplo, para habilitar el protocolo Openflow y habilitar el reenvío de paquetes se requiere de las siguientes aplicaciones:

org.onosproject.openflow **à** Habilita el protocolo openflow y apertura sus respectivos puertos (6633 y 6653) para el registro de los conmutadores.

org.onosproject.fwd **à** Habilita el reenvío de paquetes

Las aplicaciones se pueden instalar mediante CLI y WEB:

Por CLI, desde la consola de ONOS:

```
onos> app activate org.onosproject.fwd
onos> app activate org.onosproject.openflow
```

La validación de su correcta instalación se hace mediante la ejecución del siguiente comando.

```
onos> apps -a -s
```

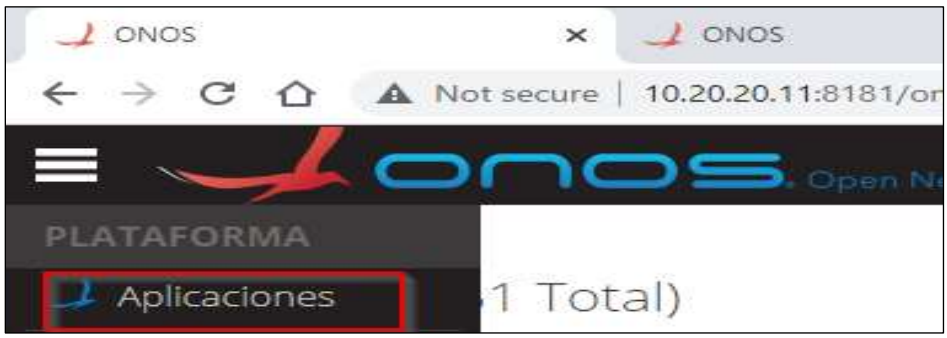
Este comando muestra el listado de todas las aplicaciones instaladas o activas en el controlador.

Por la interfaz WEB, se puede hacer de la siguiente manera.

Desplegando el menú de opción se encuentra la opción de aplicaciones

Figura 32

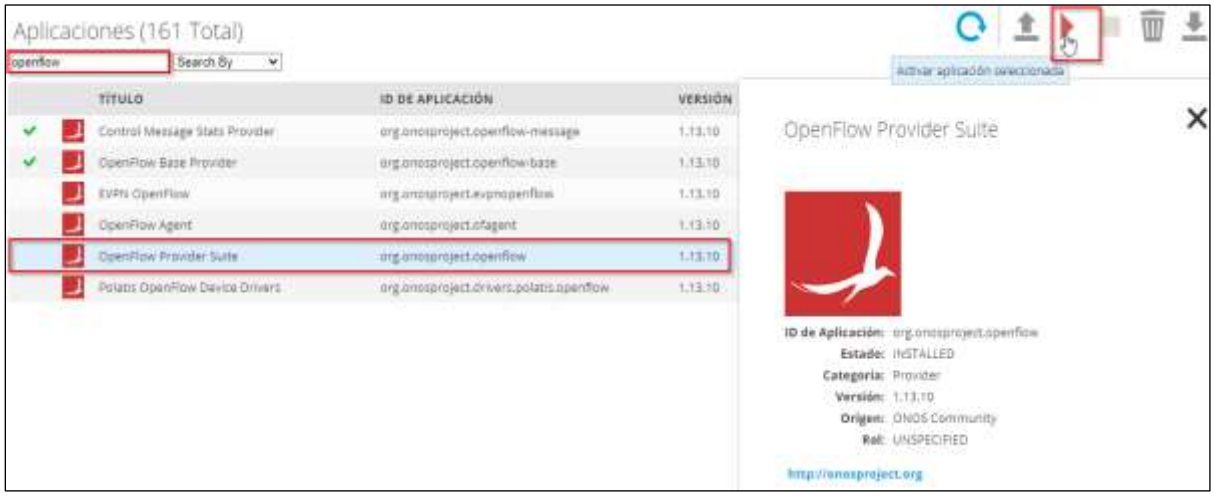
Opción aplicaciones del menú de opciones de ONOS – Interfaz WEB ONOS



En la barra de búsqueda, buscamos la aplicación a instalar. en nuestro caso la aplicación openflow, seleccionamos la aplicación y la activamos.

Figura 33

Búsqueda de aplicaciones para su instalación– Interfaz WEB ONOS



De la misma forma activamos la aplicación FWD y el resultado debe ser el siguiente:

Figura 34

Lista de aplicaciones instaladas – Interfaz WEB ONOS

4.5.4. Configurar alta disponibilidad de controlador

ONOS, para la alta disponibilidad hace uso de un cluster de controladores. Para formar el cluster, es necesario contar con mínimo 3 controladores y de esta manera formar la alta disponibilidad.

Para la formación del cluster, es necesario ejecutar el siguiente comando en cada uno de los controladores que lo conformarán.

```
~# sudo /opt/onos/bin/onos-form-cluster 10.20.20.11 10.20.20.12 10.20.20.13
```

Las direcciones IP que se muestran, corresponde a cada uno de los controladores que conformarán el cluster.

Desde la interfaz web veremos el siguiente resultado.

Figura 35

Nodos del controlador que conforman el cluster – Interfaz WEB ONOS

O desde la interfaz CLI, con el siguiente comando podemos ver los nodos que conforman el cluster.

```
onos> nodes
```

Una de las funcionalidades que ofrece ONOS mediante la alta disponibilidad es el balanceo de carga: Para activar el balanceo de carga, necesitamos de la aplicación ***org.onosproject.mlb*** y la instalación se realiza de la misma forma que en el punto anterior (4.5.4).

El resultado de instalar esta aplicación será que todos los controladores que conforman el cluster tendrán de un número de conmutadores registrados de manera equitativa.

4.5.5. Registro de los conmutadores en el controlador ONOS

Para mostrar el enrolamiento de los conmutadores en los controladores tomaremos como referencia al emulador de redes MININET.

Para hacer que la alta disponibilidad que ofrece ONOS, sea relevante y funcione, es necesario que todos los conmutadores estén registrados en los 3 controladores del cluster. Sin embargo, sólo 1 será el nodo maestro y los 2 restantes serán los controladores de contingencia.

Para el registro de los conmutadores se realiza mediante la siguiente configuración.

Vínculo entre las direcciones IP de los controladores y sus respectivos nombres

```
c0=net.addController(name='c0',
                     controller=RemoteController,
                     ip='10.20.20.11',
                     protocol='tcp',
                     port=6633)

c1=net.addController(name='c1',
                     controller=RemoteController,
                     ip='10.20.20.12',
                     protocol='tcp',
                     port=6633)

c2=net.addController(name='c2',
                     controller=RemoteController,
                     ip='10.20.20.13',
                     protocol='tcp',
                     port=6633)
```

Enrolamiento de los conmutadores en los tres conmutadores.

```
net.get('s1').start([c0,c1,c2])
net.get('s2').start([c0,c1,c2])
net.get('s3').start([c0,c1,c2])
net.get('s4').start([c0,c1,c2])
net.get('s5').start([c0,c1,c2])
net.get('s6').start([c0,c1,c2])
net.get('s7').start([c0,c1,c2])
net.get('s8').start([c0,c1,c2])
net.get('s9').start([c0,c1,c2])
net.get('s10').start([c0,c1,c2])
```

En esta porción del script de configuración de la red en mininet se puede ver, en la primera parte, el vínculo entre el nombre asignado al controlador y la dirección IP de cada uno de los controladores y en la segunda parte, está el enrolamiento en cada uno de los controladores.

Cómo se puede observar, todos los controladores tendrán como maestro al primer controlador y a los 2 restantes como contingencia ante la falla del primero (c0). Sin embargo, teniendo activado opción de balanceo de carga, estos se distribuirán de manera automática entre los 3 controladores.

4.5.6. Configuración de VLAN

Para realizar la configuración de una VLAN en el controlador ONOS, es necesario tener en cuenta el siguiente comando.

```
onos> interface-add -v 30 of:0000000000000006/2 h1
```

En donde:

- ✓ El número 30, representa el número de VLAN
- ✓ of:0000000000000006, es el nombre del conmutador
- ✓ /2, es la interface a donde se encuentra conectado el dispositivo final.
- ✓ h1, es el dispositivo final, en este caso tiene como identificados h1.

Luego usaremos la VPLS que ofrece ONOS la cual permite establecer circuitos a nivel de capa dos entre múltiples puntos finales en una red OpenFlow. En cuanto a la configuración se debe tener en cuenta lo siguiente:

- ✓ Definir la VPLS (se asigna un nombre)
- ✓ Definir las interfaces de los conmutadores que se comunicarán con los equipos que conforman parte de la VPLS.
- ✓ Asociar cada una de las interfaces a la VPLS que formarán parte.
- ✓ Tener en cuenta que deben pertenecer dos interfaces asociadas como mínimo. Así mismo, los hosts que conforman parte de la VPLS pueden tener el mismo ID de VLAN, diferente ID de VLAN, o simplemente no tenerlo.

Luego de haber configurado las interfaces se debe habilitar el VPLS (Virtual Private Lan Service) y agregar los hosts a este. Para ello, ejecutamos los siguientes el siguiente comando.

```
onos> vpls create VPLS1
onos> vpls add-if VPLS1 h1
```

En donde:

- ✓ VPLS1, en la primera línea, es el nombre con el que se creará el VPLS
- ✓ h1, es el host que se está agregando al VPLS1.

4.5.7. Script de automatización

Con la finalidad optimizar los tiempos en la ejecución de configuraciones y/o automatizar la mismas programando un horario específico y que no requiera la intervención manual del administrador de la red, se debe realizar lo siguiente:

Crear el script, al cual le daremos el nombre: **script_onos_add_int.sh**

```
# sudo vi script_onos_add_int.sh
```

Editar el script

```
#!/bin/bash

sshpass -p "rocks" ssh -o StrictHostKeyChecking=no -p 8101 -X onos@10.20.20.11 "
interface-add -v 30 of:0000000000000006/2 h1;
vpls create VPLS1;
vpls add-if VPLS1 h1"
```

Instalar sshpass, esta aplicación es necesario para poder ejecutar la conexión con SSH, y lo hacemos mediante el siguiente comando.

```
# sudo apt-get install sshpass
```

Permisos de ejecución al script creado.

```
# sudo chmod +x script_onos_add_int.sh
```

Ejecutar el script.

```
# ./script_onos_add_int.sh
```

Automatizar la ejecución del script.

Para realizar la automatización en la ejecución de nuestro script, utilizaremos el administrador regular de procesos de nuestro sistema operativo Ubuntu sobre el cual está instalado el controlador ONOS, Cron. Cron administra los procesos en segundo plano (demonio) que ejecuta procesos a intervalos regulares (por ejemplo, cada minuto, día, semana o mes) haciendo uso de la tabla de tareas crontab.

Editar crontab:

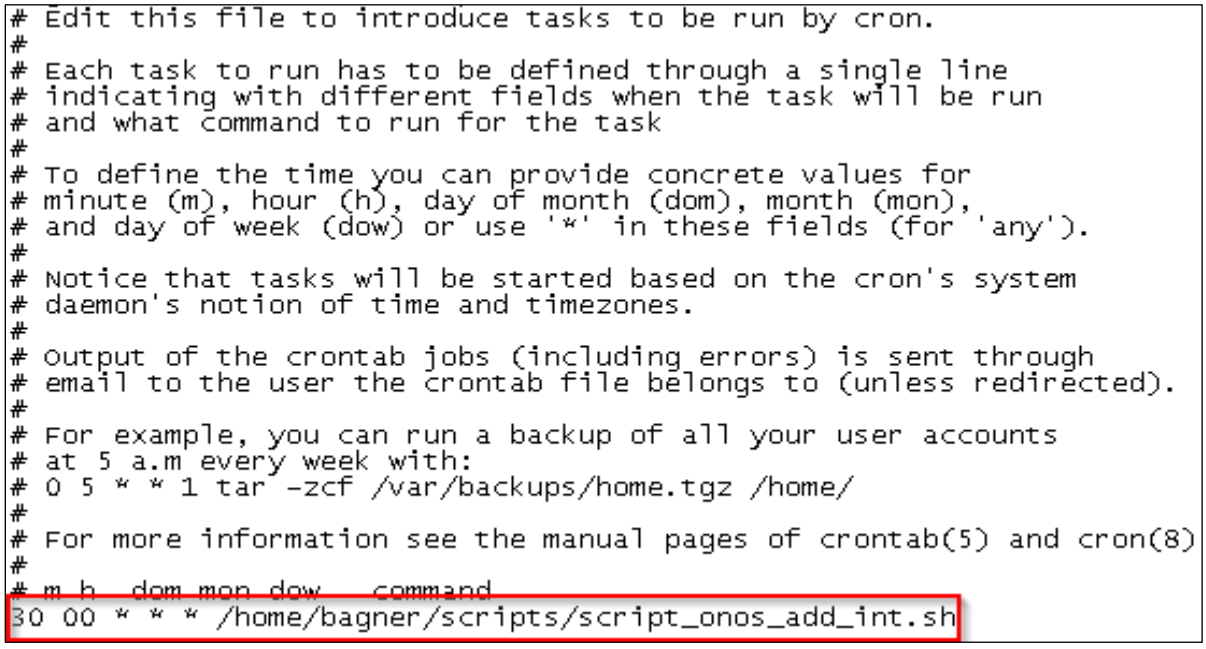
```
# crontab -e
```

Con el comando ***crontab –e*** iniciamos el editor en donde configuraremos la hora a ejecutar de nuestro script. En nuestro caso, a modo de ejemplo, configuraremos para que el script se ejecute a las 00:30 horas de siguiente día.

```
00 30 * * * /home/bagner/scripts/script_onos_add_int.sh
```

Figura 36

Consulta de tareas programad en crontab - #crontab -l



Ejecutando el comando crontab –l, se visualiza todas las tareas que se ejecutarán de manera automática de acuerdo a la programación que el administrador de red haya realizado. En la figura 36 se muestra que el script ***script_onos_add_int.sh*** se ejecutará a las 00:30 horas del día siguiente.

5. CAPÍTULO 5: PRUEBAS

5.1. Escenario de pruebas

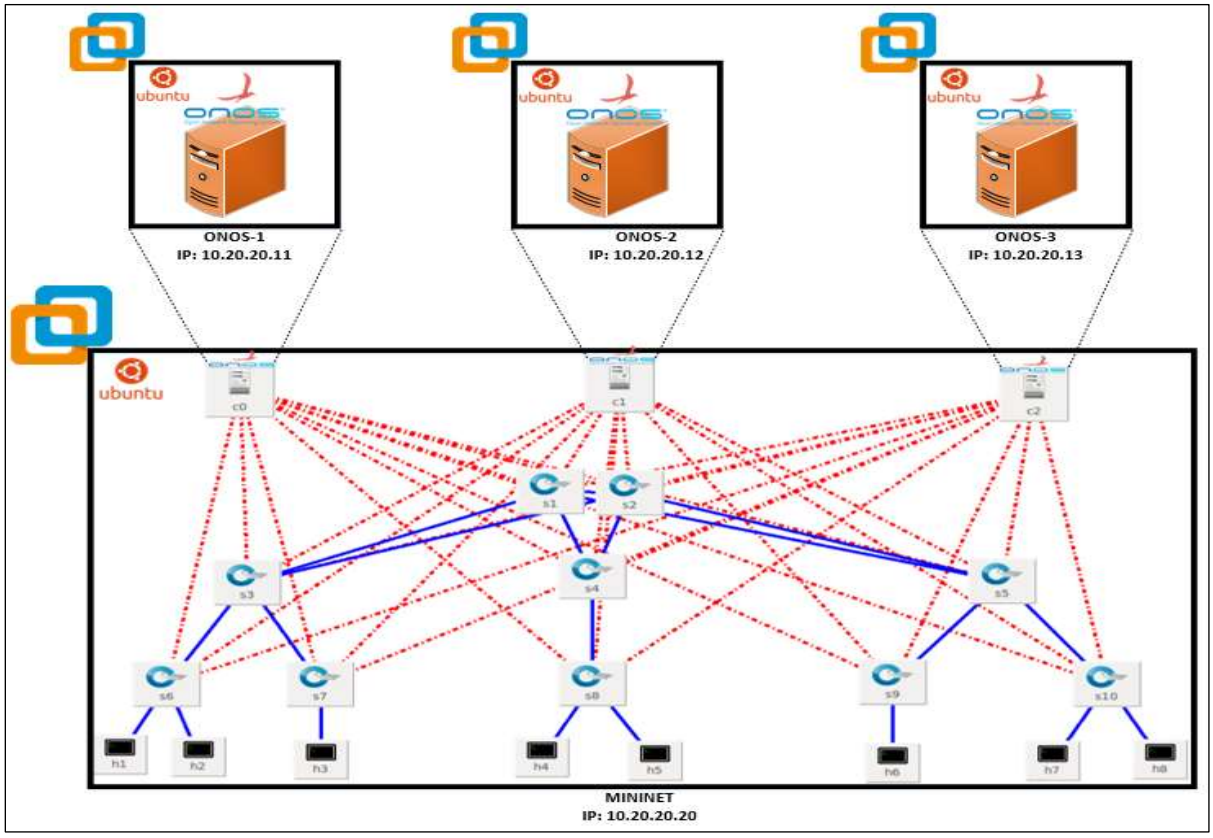
Para llevar a cabo las pruebas y la validación del proyecto, nos ayudaremos de las siguientes herramientas:

- **VMware Workstation:** Esta herramienta de virtualización se ha utilizado para virtualizar las 4 máquinas virtuales (3 controladores y mininet).
- **Sistema Operativo Ubuntu, versión 20.04:** Sistema Operativo sobre el cual se ha implementado los controladores ONOS y el emulador mininet
- **MININET, versión 2.2.2**: Emulador de redes SDN, sobre el cual emularemos la red SDN.
- **Controlador ONOS, versión 1.13.10:** Este controlador estará configurado en alta disponibilidad, formando un cluster de 3 nodos

En la siguiente imagen se puede observar un diagrama que describe las 4 máquinas virtuales:

Figura 37

Escenario de pruebas



De acuerdo con el diagrama de la figura 37:

- La máquina virtual ONOS-1 tiene instalado el sistema operativo Ubuntu 20.04, sobre el cual se ha implementado el primer controlador SDN ONOS. La dirección IP asignada es 10.20.20.11.
- La máquina virtual ONOS-2 tiene instalado el sistema operativo Ubuntu 20.04, sobre el cual se ha implementado el segundo controlador SDN ONOS. La dirección IP asignada es 10.20.20.12
- La máquina virtual ONOS-3 tiene instalado el sistema operativo Ubuntu 20.04, sobre el cual se ha implementado el tercer controlador SDN ONOS. La dirección IP asignada es 10.20.20.13
- La máquina virtual MININET tiene instalado el sistema operativo Ubuntu 20.04, sobre el cual se ha implementado el emulador mininet. La dirección IP asignada es 10.20.20.20

5.1.1. Simulación de la topología de pruebas

La topología de red, sobre la cual realizaremos las pruebas, es una representación mínima, respecto de la red original del proyecto. Es posible realizar una representación mínima debido a las siguientes razones:

- ✓ Conmutadores Principales, estos conmutadores forman parte de la simulación, debido a que son los equipos que concentran las conexiones de todos los conmutadores secundarios.
- ✓ Se consideró únicamente 3 conmutadores secundarios (conmutadores de distribución) debido a que todos los conmutadores de la red original del proyecto tienen funciones parecidas, es decir, tienen configuraciones que en algunos casos se repiten y en otros son muy idénticas.
- ✓ Se consideró únicamente 8 conmutadores de accesos como una representación mínima de la red original del proyecto, esto debido a las mismas razones del punto anterior.

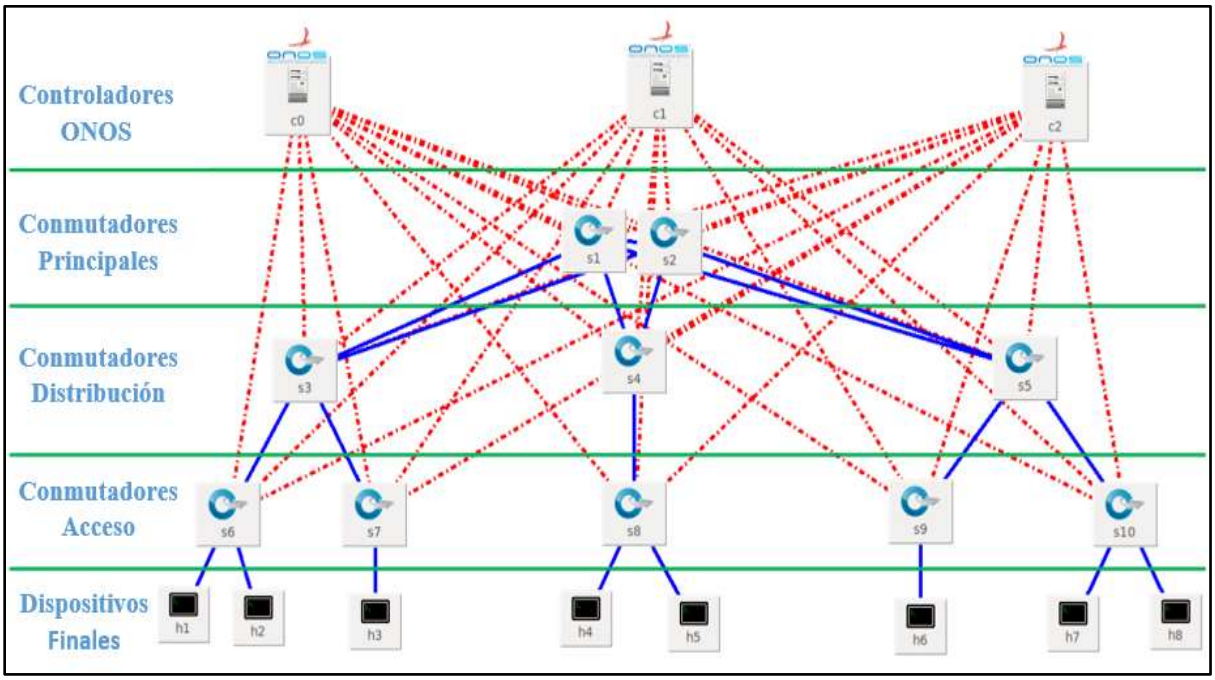
Dispositivos que conforman la red a simular:

- ✓ **3 controladores ONOS** (los mismos que forman un cluster para hacer que la red sea tolerante a fallos por parte del controlador)

- ✓ **2 conmutadores principales,** estos conmutadores cumplen la función de concentrar a todos los conmutadores secundarios, los mismos que están instalados en cada uno de los pabellones y en los diferentes pisos de estos.
- ✓ **3 conmutadores secundarios o de distribución,** estos concentradores son los que concentran a los conmutadores finales.
- ✓ **8 conmutadores de acceso,** son los conmutadores encargados de brindar el servicio de red a los usuarios y dispositivos finales.
- ✓ **12 dispositivos finales,** los mismos que simularán la función que cumplen las computadoras, teléfonos, impresoras, proyectores, etc.

Figura 38

Diagrama de red creada en el simulador Mininet



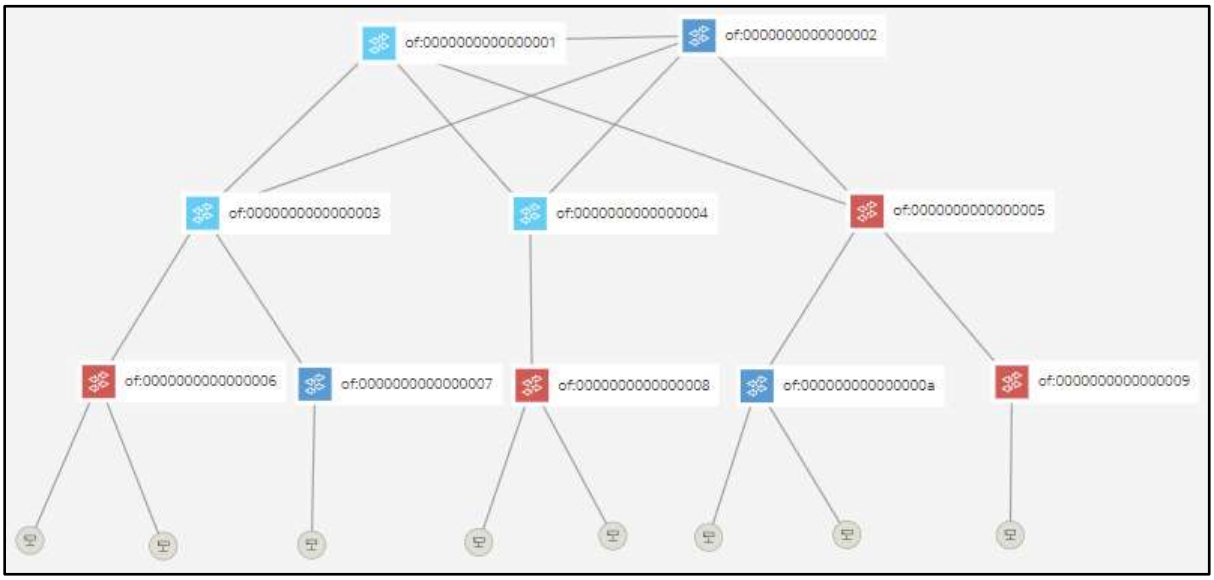
5.1.2. Topología de pruebas en el controlador ONOS

La red creada en el simulador Mininet, está integrada con el controlador ONOS Nightingale (Ruiseñor), los mismo que serán los encargados de gestionar todos los conmutadores.

En la siguiente figura se muestra la topología que se visualiza en el entorno web del controlador ONOS.

Figura 39

Topología de red según el controlador ONOS



5.1.1. Scrip de la topología creada en Mininet

El script mediante el cual se hace la conexión de cada uno de los conmutadores hacia el controlador ONOS es el siguiente:

A continuación, se describe las partes más importantes del script.

```
#####agregado de controladores, se detalla el nombre, ip, puerto#########
   c1=net.addController(name='c1',
                        controller=RemoteController,
                        ip='10.20.20.12',
                        protocol='tcp',
                        port=6633)

#####agregado de conmutadores, esta línea se repetirá tantas veces cómo la cantidad
de conmutadores que conforman la red#########
   s1 = net.addSwitch('s1', cls=OVSKernelSwitch)

#####agregado de host, esta línea se repetirá tantas veces cómo la cantidad de host
que conforman la red#########
   h1 = net.addHost('h1', cls=Host, ip='10.0.0.1', defaultRoute=None)

#####agregado de enlaces, #########
   net.addLink(s1, s2)

##### Sentencia para iniciar la red#########
   net.build()

#####Sentencia para iniciar los controladores #########
   for controller in net.controllers:
      controller.start()
```

```
#####Sentencia para iniciar los conmutadores #########
info( '*** Starting switches\n')
   net.get('s10').start([c0,c1,c2])
```

5.2. Gestión Centralizada de los conmutadores de la red

En este punto se demostrará el objetivo específico número 1 (Realizar un diseño de red que centralice la gestión de los dispositivos de capa 2 del modelo OSI (conmutadores) en el controlador)

Cómo se ha mostrado en el diagrama del escenario de pruebas, el controlador ONOS es el encargado de centralizar toda la gestión de los conmutadores.

PRUEBAS

Cómo parte de las validaciones, en busca de demostrar la gestión centralizada, se realizará las siguientes pruebas:

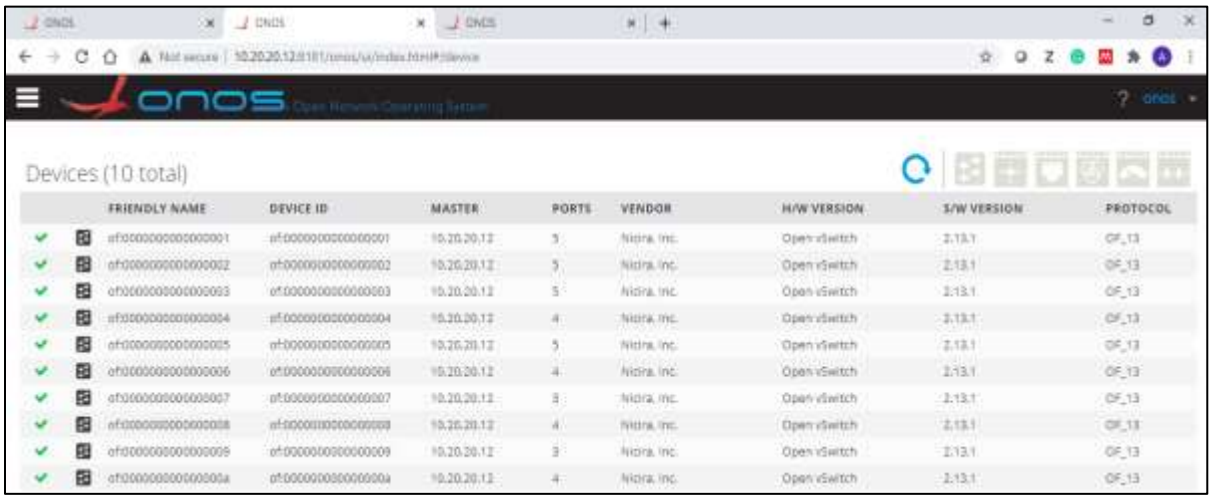
- ✓ Inventario de Conmutadores
- ✓ Inventario de host
- ✓ Eliminar conmutadores y hosts desde el controlador
- ✓ Cambio de modo maestro en los conmutadores

5.2.1. Inventario de los conmutadores

Desde el controlador se pueden visualizar todos los conmutadores enrolados y ver información importante (puertos, fabricante, versión, nombre, etc.) por cada uno de ellos.

Figura 40

Lista de conmutadores registrados



En la figura 40, se puede ver el inventario de cada uno de los conmutadores registrados en el controlador, así como información específica por cada uno de ellos:

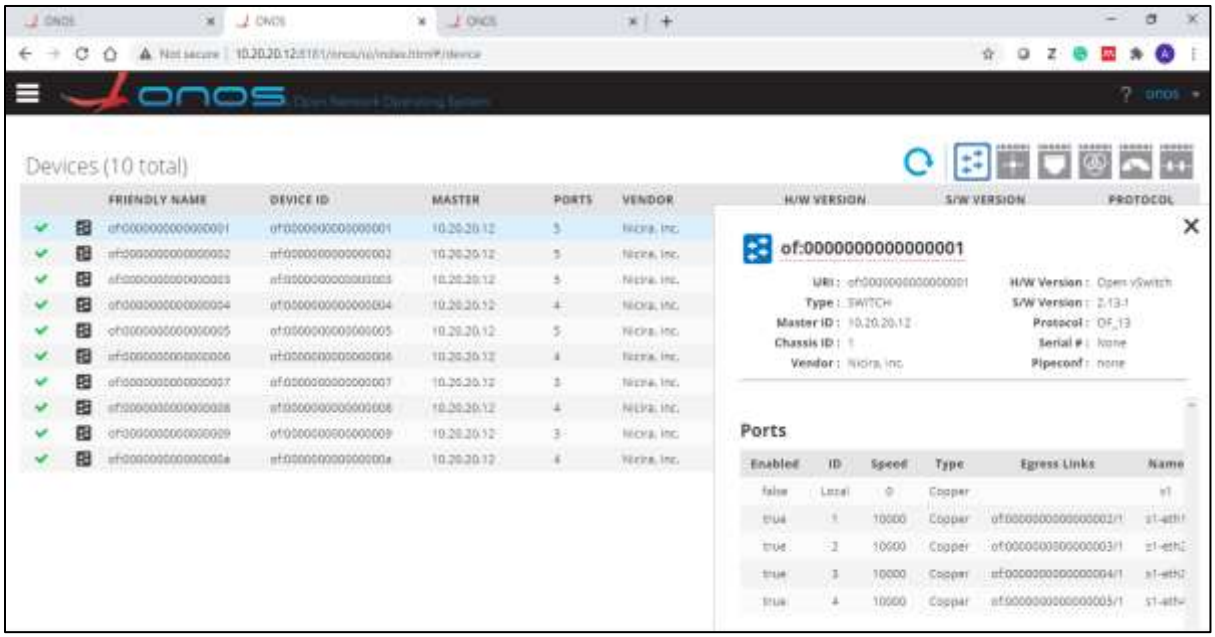
- ✓ **FRIENDLY NAME:** este nombre se puede configurar desde el mismo controlador de una manera amigable para identificar a cada uno los dispositivos.
- ✓ **Device ID:** Es un código asignado por el controlador para identificar a cada uno de los conmutadores.
- ✓ **Master:** identifica el controlador por el cual está gestionado, en nuestro caso se puede ver 3 diferentes direcciones IP, las mismas que corresponden a los nodos que componen el cluster.
- ✓ **VENDOR:** Muestra el fabricante del conmutador
- ✓ **H/W VERSION:** Versión del hardware del controlador
- ✓ **S/W VERSION:** Versión del Software que tiene el conmutador
- ✓ **PROTOCOL:** Versión del protocolo OpenFlow.

Así mismo, si se requiere ver el detalle de un conmutador en específico; como, por ejemplo, los puertos, velocidad de puertos, nombre de las interfaces, etc. se puede visualizar seleccionando el conmutador en cuestión.

Para demostrar esta funcionalidad que tiene el controlador ONOS, elegiremos el conmutador que tiene ID: **of:0000000000000001**

Figura 41

Detalles del conmutador of:0000000000000001



Haciendo uso de la opción "***show flow view for this device***", se puede ver el tráfico que pasa a través de este conmutador.

Figura 42

Opciones del conmutador of:0000000000000001

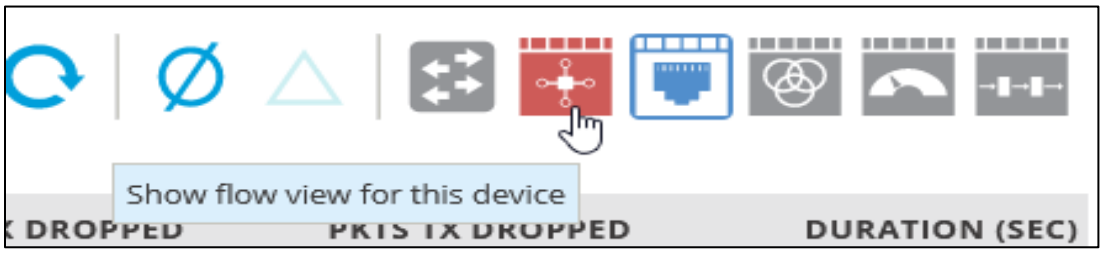
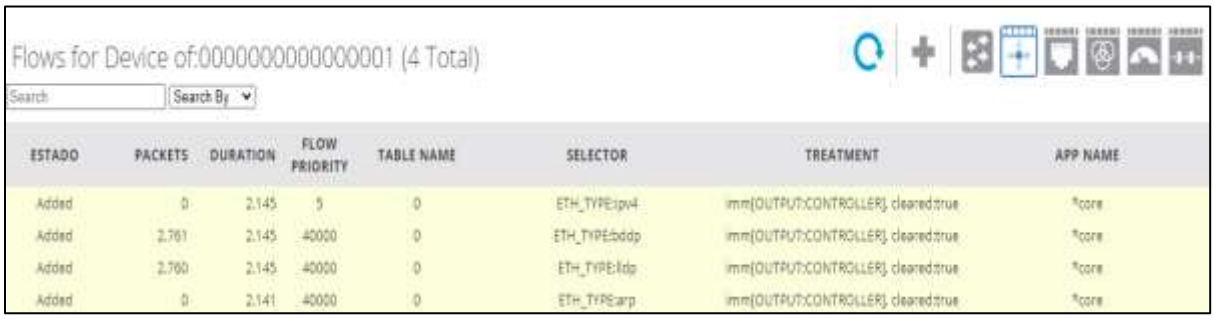


Figura 43

Tráfico que pasa por el conmutador of:0000000000000001

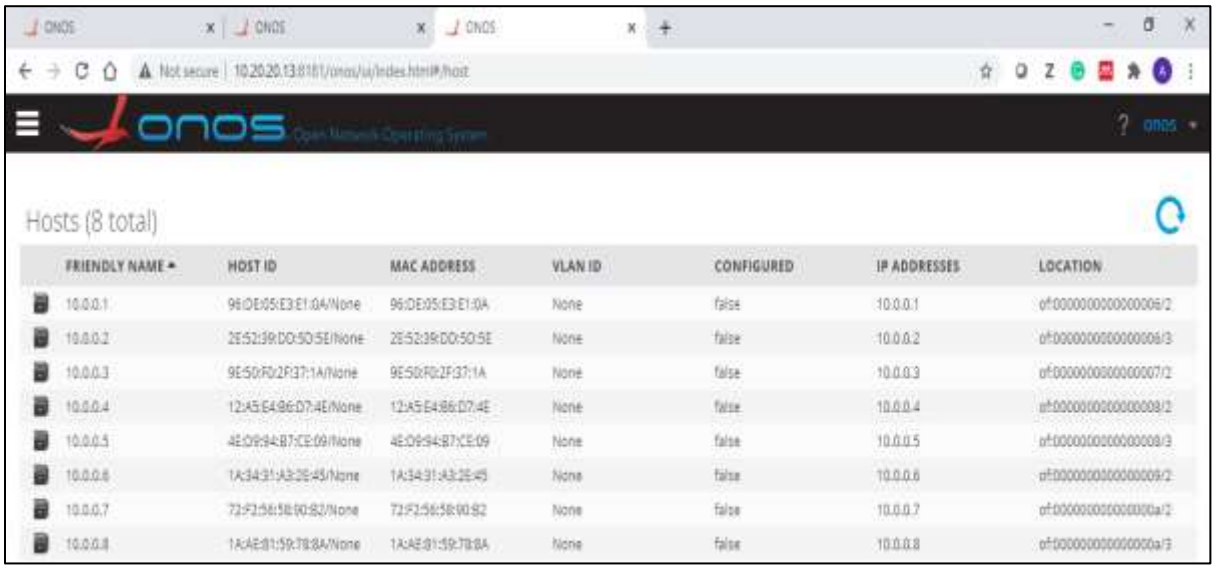


5.2.2. Inventario de los Host

El controlador ONOS, también tiene la capacidad de listar todos los hosts (dispositivos finales) que están dentro de la red. Es importante que estos dispositivos estén activos.

Figura 44

Lista de los hosts que conforman la red (host activos)



Así como en el caso de los conmutadores, en esta opción podemos ver información muy importante por cada de uno de los hosts como son la dirección IP, dirección MAC, VLAN, conmutador al que se encuentran conectados, etc.

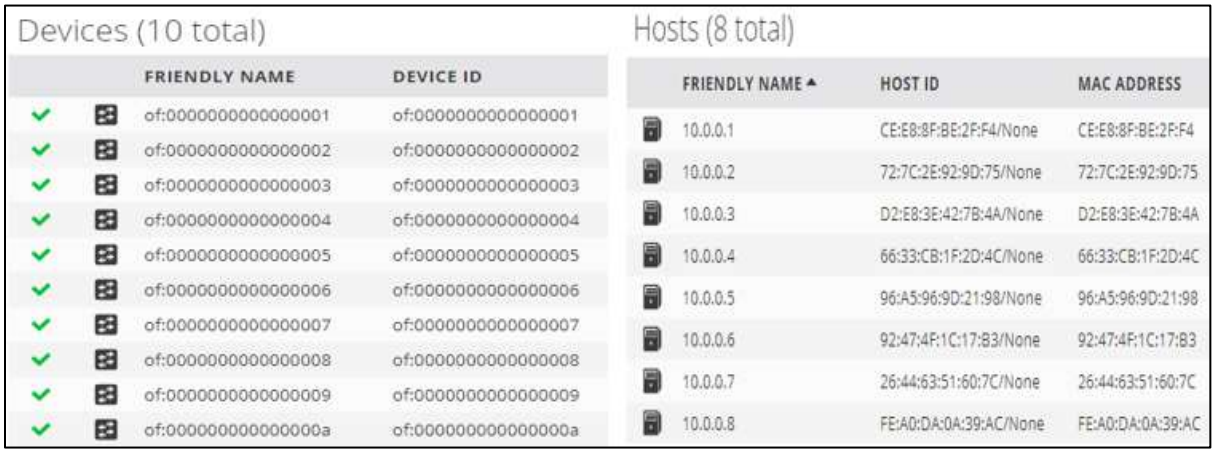
5.2.3. Eliminar conmutadores y hosts desde el controlador

Esta prueba consiste en poder eliminar a uno o más conmutadores o host que están registrados en el controlador y que por alguna razón ya no forman parte de la red, debemos estar en la capacidad de poder eliminarlos también del controlador.

Cómo vemos en la siguiente imagen, existen 10 conmutadores y 8 host.

Figura 45

Lista de los conmutadores y host que forman parte de la red



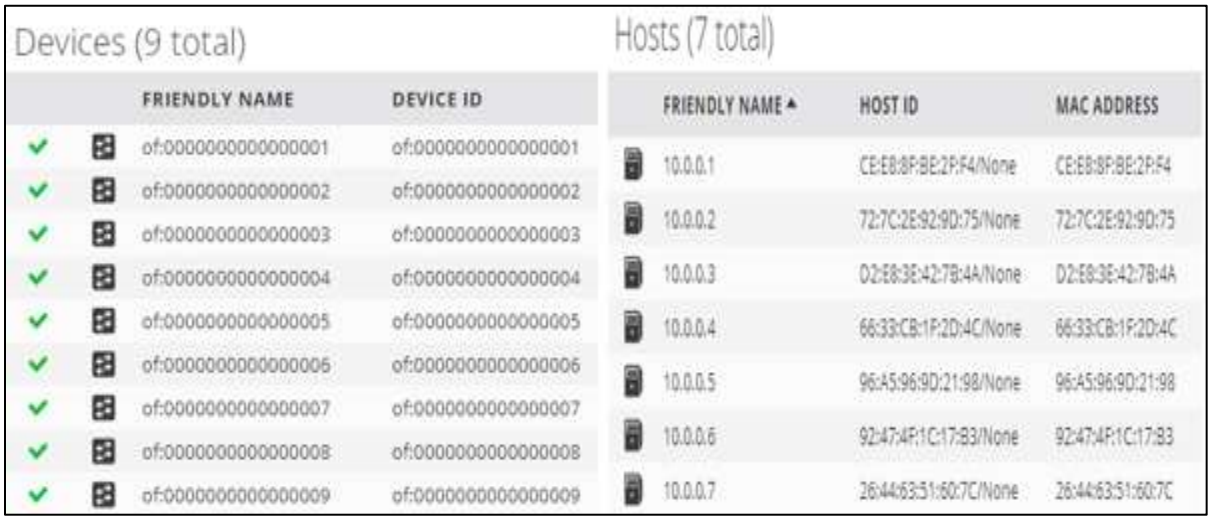
Con los siguientes comandos, por CLI, eliminaremos un conmutador y un host para lo cual, elegiremos el último equipo de cada una de las listas.

```
onos:device-remove of:000000000000000a
onos:host-remove FE:A0:DA:0A:39:AC/None
```

Como podemos observar en la siguiente imagen, en cada una de las listas a disminuido un equipo, el último de cada una de las listas ha sido eliminado. En el caso de los conmutadores se ha eliminado al conmutador ***of:000000000000000a*** y en el caso de los hosts, se ha eliminado a host ***FE:A0:DA:0A:39:AC/None***.

Figura 46

Lista de los conmutadores y host que forman parte de la red, luego de eliminar uno de cada lista



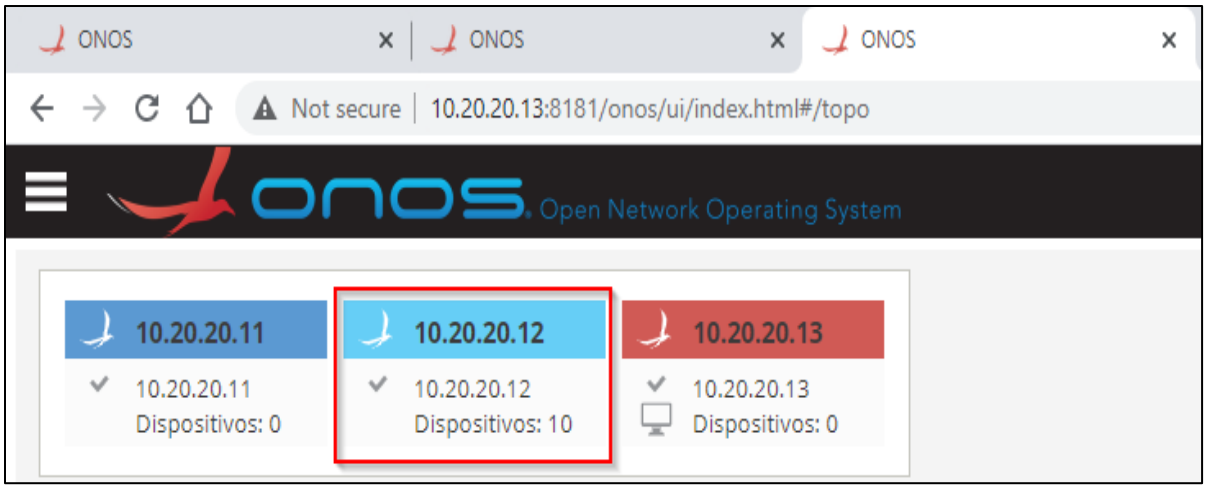
5.2.4. Cambio de nodo maestro en los controladores

Cómo bien se mencionó en los puntos anteriores, para que exista alta disponibilidad es necesario que los conmutadores se conecten a todos los controladores que conforman el cluster, en nuestro caso 3 nodos.

Para realizar esta prueba, la función de balanceo de carga entre los nodos del cluster la tendremos deshabilitada. En este sentido, cuando se inicia la red, los conmutadores se enrolan a cualquiera de los nodos de manera aleatoria pudiendo estar enrolados; todos, a un solo controlador, tal como se muestra en la siguiente imagen, todos los controladores se encuentran dominados por el nodo 10.20.20.12.

Figura 47

Nodos del cluster, nodo 10.20.20.12 con 10 conmutadores enrolados



Lo siguiente que realizaremos, mediante comandos por CLI, es:

- ✓ Los conmutadores s1 (of:0000000000000001), s2 (of:0000000000000002), s3 (of:0000000000000003) y s4 (of:0000000000000004) tengan como maestro al nodo 10.20.20.11.

- ✓ Los conmutadores s5 (of:0000000000000005), s6 (of:0000000000000006) y s7 (of:0000000000000007) tengan como nodo maestro al nodo 10.20.20.12

- ✓ Los conmutadores s8 (of:0000000000000008), s9 (of:0000000000000009) y s10 (of:000000000000000a) tengan como nodo maestro al nodo 10.20.20.13.

En la siguiente imagen veremos que el nodo maestro para todos los controladores en el que tiene IP: 10.20.20.12, esta es la muestra inicial, antes de realizar cambios.

Figura 48

Lista de conmutadores con su respectivo nodo maestro, antes de los cambios planteados

```
onos> onos:masters
10.20.20.11: 0 devices
10.20.20.12: 10 devices
  of:0000000000000001
  of:0000000000000002
  of:0000000000000003
  of:0000000000000004
  of:0000000000000005
  of:0000000000000006
  of:0000000000000007
  of:0000000000000008
  of:0000000000000009
  of:000000000000000a
10.20.20.13: 0 devices
```

Primero, con los siguientes comandos registraremos los conmutadores s1, s2, s3 y s4 al nodo 10.20.20.11

```
onos> onos:device-role of:0000000000000001 10.20.20.11 master
onos> onos:device-role of:0000000000000002 10.20.20.11 master
onos> onos:device-role of:0000000000000003 10.20.20.11 master
onos> onos:device-role of:0000000000000004 10.20.20.11 master
```

Figura 49

Lista de conmutadores con su respectivo nodo maestro, después del primer cambio

```
onos> onos:masters
10.20.20.11: 4 devices
  of:0000000000000001
  of:0000000000000002
  of:0000000000000003
  of:0000000000000004
10.20.20.12: 6 devices
  of:0000000000000005
  of:0000000000000006
  of:0000000000000007
  of:0000000000000008
  of:0000000000000009
  of:000000000000000a
10.20.20.13: 0 devices
```

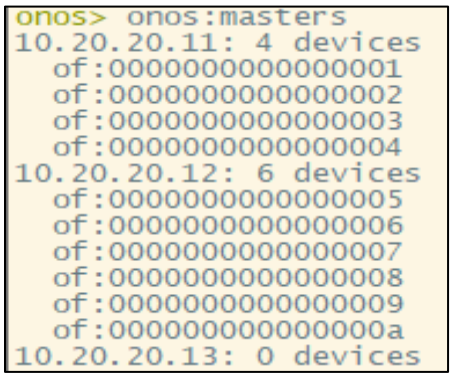
Tal como se aprecia en la imagen, el primer nodo ya tiene registrado a los 4 primeros conmutadores.

Segundo, con los siguientes comandos registraremos los conmutadores s5, s6 y s7 al nodo 10.20.20.12

```
onos> onos:device-role of:0000000000000005 10.20.20.12 master
onos> onos:device-role of:0000000000000006 10.20.20.12 master
onos> onos:device-role of:0000000000000007 10.20.20.12 master
```

Figura 50

Lista de conmutadores con su respectivo nodo maestro, después del segundo cambio



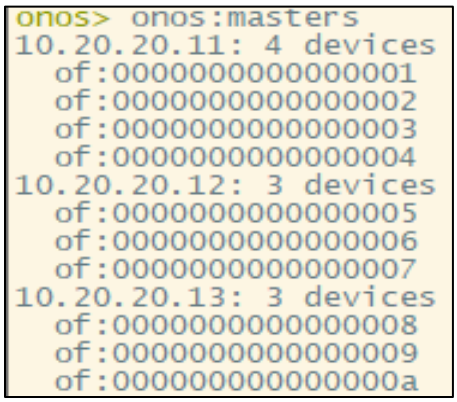
En la figura 50, podemos observar lo mismo que en la figura 49, debido a que el segundo nodo tenía inicialmente registrado a todos los conmutadores, prácticamente se está reconfirmando el enrolamiento para el nodo 2.

Finalmente, registramos los conmutadores s8, s9y s10 al nodo 10.20.20.13

```
onos> onos:device-role of:0000000000000008 10.20.20.13 master
onos> onos:device-role of:0000000000000009 10.20.20.13 master
onos> onos:device-role of:000000000000000a 10.20.20.13 master
```

Figura 51

Lista de conmutadores con su respectivo nodo maestro, después del segundo cambio



El resultado final, luego de los 3 pasos, es el registro de cada uno de los conmutadores de acuerdo a lo planteado.

Figura 52

Lista de conmutadores con su respectivo nodo maestro, después del segundo cambio

```
onos> onos:roles
of:0000000000000001: master=10.20.20.11, standbys=[ 10.20.20.11 10.20.20.12 10.20.20.13 ]
of:0000000000000002: master=10.20.20.11, standbys=[ 10.20.20.11 10.20.20.12 10.20.20.13 ]
of:0000000000000003: master=10.20.20.11, standbys=[ 10.20.20.11 10.20.20.12 10.20.20.13 ]
of:0000000000000004: master=10.20.20.11, standbys=[ 10.20.20.11 10.20.20.12 10.20.20.13 ]
of:0000000000000005: master=10.20.20.12, standbys=[ 10.20.20.12 10.20.20.13 10.20.20.11 ]
of:0000000000000006: master=10.20.20.12, standbys=[ 10.20.20.12 10.20.20.13 10.20.20.11 ]
of:0000000000000007: master=10.20.20.12, standbys=[ 10.20.20.12 10.20.20.13 10.20.20.11 ]
of:0000000000000008: master=10.20.20.13, standbys=[ 10.20.20.13 10.20.20.12 10.20.20.11 ]
of:0000000000000009: master=10.20.20.13, standbys=[ 10.20.20.13 10.20.20.12 10.20.20.11 ]
of:000000000000000a: master=10.20.20.13, standbys=[ 10.20.20.13 10.20.20.11 10.20.20.12 ]
```

Adicionalmente, la figura 52, muestra a todos los conmutadores registrados con sus respectivos nodos maestros y los nodos secundarios ante la falla del nodo maestro.

RESULTADO DE LAS PRUEBAS

Finalmente, el resultado de las pruebas realizadas en este punto es satisfactorio

Tabla 14

Resultado de pruebas cambio de nodo maestro en los controladores

PRUEBA	RESULTADO	COMENTARIOS
Inventario de Conmutadores	✓	Se logró evidenciar que el controlador ONOS permite realizar un inventario de todos los conmutadores enrolados en cada uno de los nodos que conforman el cluster
Inventario de host	✓	Se logró evidenciar que el controlador ONOS permite realizar un inventario de todos los hosts que se conectan a cada uno de los conmutadores enrolados.
Eliminar conmutadores o hosts desde el controlador	✓	Se ha logrado demostrar que el controlador permite eliminar conmutadores y hosts desde la interfaz de línea de comandos.
Cambio de modo maestro en los conmutadores	✓	Se logró evidenciar que, desde el controlador, por línea de comandos, se puede realizar el cambio de nodo maestro.

5.3. Alta disponibilidad y balanceo de carga entre los nodos del controlador

En este punto se demostrará el objetivo específico número 2 (Diseñar un arreglo de controladores SDN que tenga alta disponibilidad, de tal manera que los conmutadores no pierdan conectividad con el controlador)

Para lograr que el servicio del controlador sea tolerante a fallas, se ha contemplado en el diseño 3 nodos de ONOS, los mismos que formarán un cluster.

Y para cumplir con esta consideración, tal como se menciona en el Capítulo 4, todos los conmutadores tienen que estar registrados en todos los nodos que conforman el cluster.

En la figura 53 se puede ver una porción del script de la red implementada para la simulación y pruebas. Con la instrucción ***net.get*** se inicia la conexión de cada uno de los conmutadores con cada uno de los nodos del controlador.

Figura 53

Conexión de los conmutadores a cada uno de los controladores

```
net.get('s7').start([c0,c1,c2])
net.get('s9').start([c0,c1,c2])
net.get('s8').start([c0,c1,c2])
net.get('s1').start([c0,c1,c2])
net.get('s6').start([c0,c1,c2])
net.get('s3').start([c0,c1,c2])
net.get('s4').start([c0,c1,c2])
net.get('s2').start([c0,c1,c2])
net.get('s10').start([c0,c1,c2])
net.get('s5').start([c0,c1,c2])
```

En la figura 53, se puede ver claramente que todos los conmutadores están conectados a los 3 nodos del cluster ONOS que tienen como nombre C0, C1 y C2.

Para las siguientes pruebas, y con el propósito de mantener un mejor orden, llamaremos a cada uno de los controladores ONOS que conforman el cluster de acuerdo con la siguiente tabla.

Tabla 15

Relación nombre, IP y Nodos de los controladores ONOS utilizados

Máquina Virtual Ubuntu	Nombre del Controlador	Nodo N°	Dirección IP
ONOS-1	C0	1	10.20.20.11
ONOS-2	C1	2	10.20.20.12
ONOS-3	C2	3	10.20.20.13

PRUEBAS

Cómo parte de las validaciones, en busca de demostrar la alta disponibilidad de los controladores, se realizará las siguientes pruebas:

- ✓ Formación del cluster
- ✓ Balanceo de carga
- ✓ Continuidad del servicio ante la falla de un nodo

5.3.1. Formación del Cluster

El cluster será conformado por 3 nodos de ONOS versión 1.13.10.

Para la conformación del cluster, luego de haber realizado una correcta instalación y configuración del controlador será necesario ejecutar el siguiente comando en cada uno de los nodos.

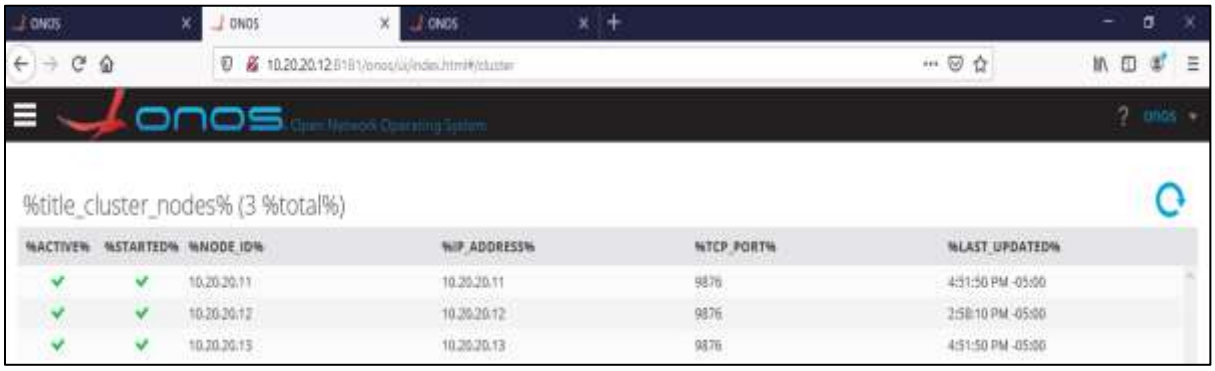
```
$sudo onos/bin/onos-form-cluster 10.20.20.11 10.20.20.12 10.20.20.13
```

Las direcciones IPs que se visualizan, corresponden a la dirección IP de cada uno de los controladores.

Luego de haber ejecutado el comando descrito, el resultado debe ser el siguiente:

Figura 54

Lista de los nodos ONOS que conforman el cluster



En la figura 54 se puede observar todos los nodos que conforman el cluster, en nuestro caso se visualiza los 3 nodos implementados.

Figura 55

Lista de los nodos ONOS que conforman el cluster, con la cantidad de conmutadores registrados

En la figura 55, hemos ingresado en la opción topología y nos muestra los 3 nodos y la cantidad de conmutadores (dispositivos) conectados a cada uno de ellos.

5.3.2. Balanceo de Carga

Para lograr que los controladores tengan distribuida la carga, ONOS tiene la opción de hacer un balanceo de carga entre los nodos que conforman el cluster; para ello, es necesario activar la siguiente aplicación: ***Mastership Load Balancer***.

Para demostrar su correcto funcionamiento haremos lo siguiente:

- ✓ Iniciaremos la prueba con los 3 nodos activos, ver figura 56.

Figura 56

Lista de los nodos de cluster ONOS operativos

Nodos del Cluster (3 Total)

ACTIVO	INICIADO	ID DEL NODO	DIRECCIÓN IP
✓	✓	10.20.20.11	10.20.20.11
✓	✓	10.20.20.12	10.20.20.12
✓	✓	10.20.20.13	10.20.20.13

Antes de ejecutar el balanceo de carga, podemos ver que todos los conmutadores tienen como nodo maestro al controlador C1 (figura 57).

Figura 57

Lista de los nodos de cluster ONOS operativos

Luego procederemos a activar la aplicación ***Mastership Load Balancer,*** lo podemos hacer desde la interfaz web o con la siguiente línea de comandos:

```
onos> app activate org.onosproject.mlb
```

Es importante tener en cuenta que este comando se ejecuta únicamente en uno de los nodos, no es necesario ejecutar el comando en los 3 nodos.

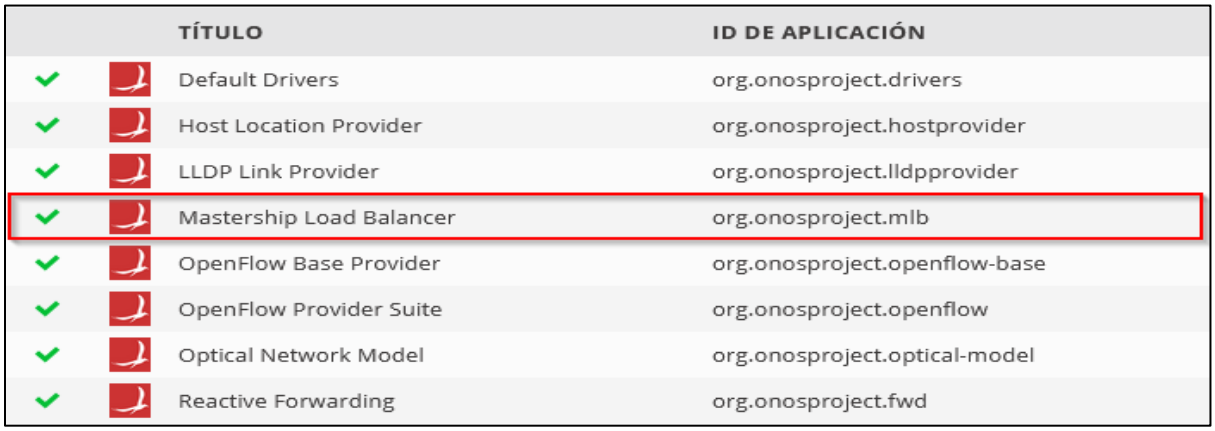
El resultado lo podemos validar con el siguiente comando:

```
apps -a -s
```

Este comando mostrará todas las aplicaciones que están instaladas. Otra manera de visualizar el resultado es por la interfaz web. El resultado debe ser tal como se muestra en la figura 58.

Figura 58

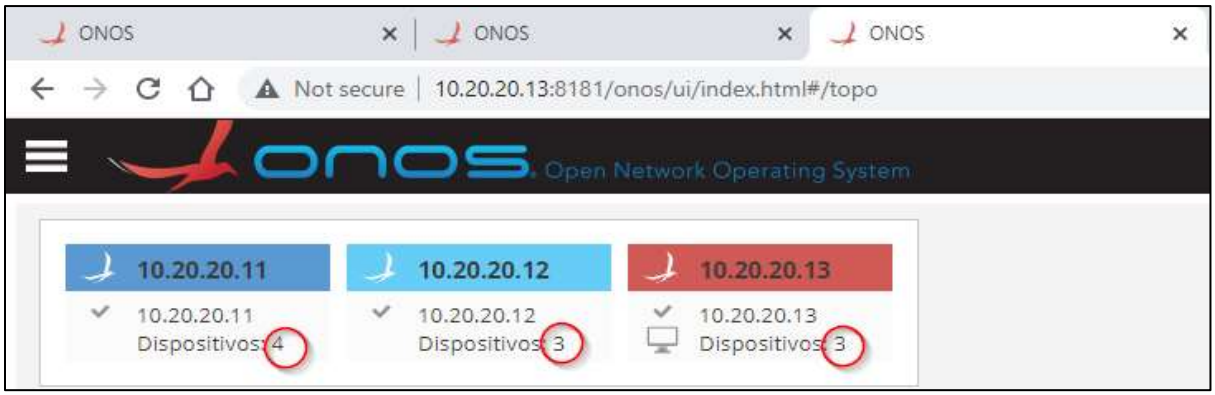
Aplicaciones activas en el controlador – Aplicación Matership Load Balancer activa



Ahora ya tenemos instalado la aplicación ***Mastership Load Balancer***, tal como se esperaba, cada uno de los nodos tienen de manera distribuida el registro de los conmutadores.

Figura 59

Carga distribuida entre los 3 nodos del cluster



Para validar el funcionamiento del balanceo de carga, ahora simularemos una falla del controlador C0, para ello desconectaremos la red desde el software de virtualización lo cual hará que el controlador ya no esté disponible, y lo que se espara es que los conmutadores se distribuyan entre el controladro C1 y C2.

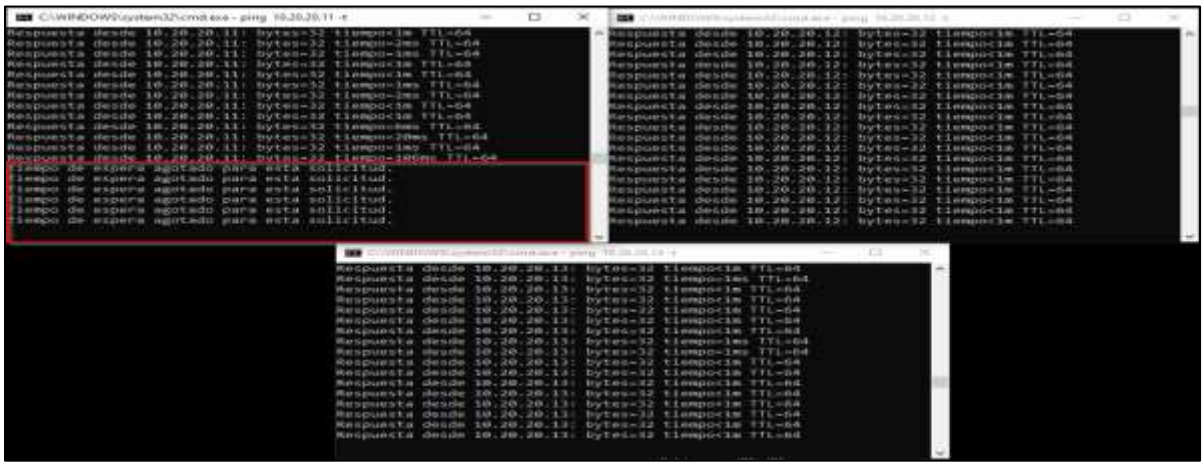
Figura 60

Desconexión de la tarjeta de red del controlador C0

Luego de haber desconectado la tarjeta de red del contrlador, podemos ver en la figura 61 que el ping ya no responde, lo cual indica que el controlador ya no se enceuntra disponible.

Figura 61

Prueba de conectividad hacia los 3 nodos del cluster, el primer nodo ya no responde



Con el controlador C0 fuera de servicio, todos los conmutadores se han distribuido entre los controladores C1 y C2; tal como se muestra en la figura 62.

Figura 62

Lista de los nodos ONOS – 1er nodo fuera de línea – Carga distribuida entre el 2do y 3er nodo

5.3.3. Prueba de falla de un nodo del Cluster

La siguiente prueba consistirá en dejar fuera de servicio a un controlador.

Para ello, primero hemos conectado la red del controlados C0 de tal manera que todos los controladores estén activos, tal como en la figura 55.

Luego, desconectamos la red del controlador C1 para simular una falla del controlador que tiene registrado a 3 conmutadores.

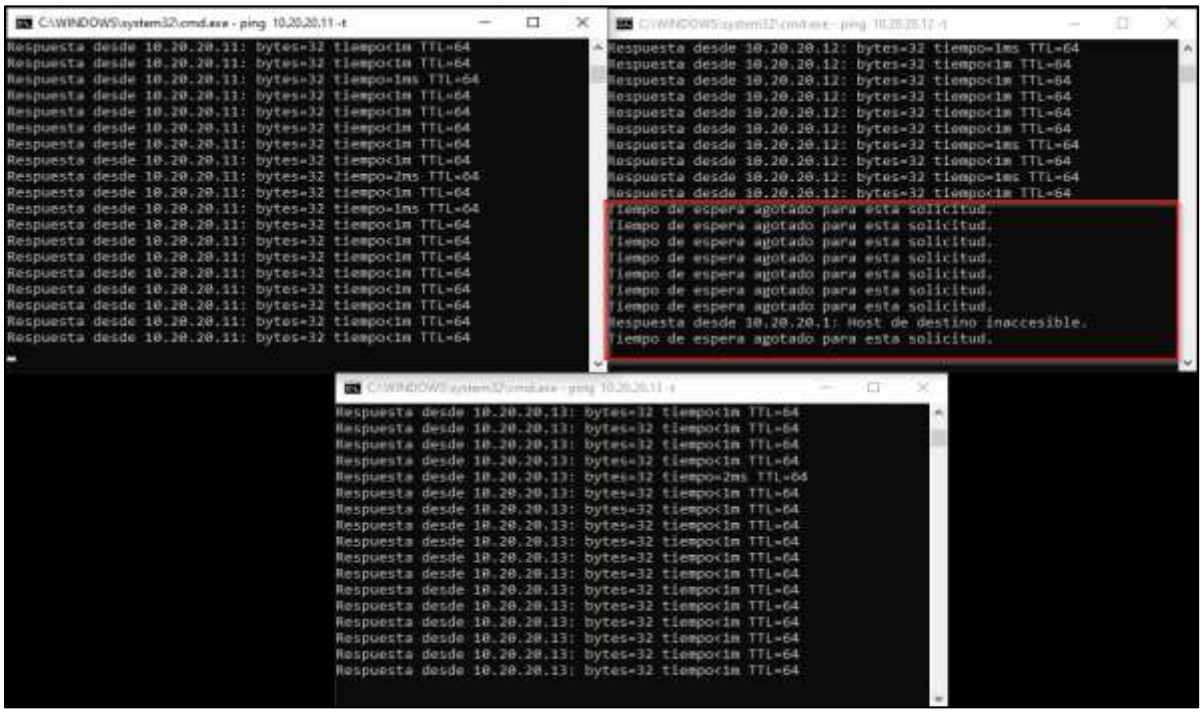
Figura 63

Desconexión de la tarjeta de red del controlador C1

Tal como se puede apreciar en la siguiente figura 64, la IP:10.20.20.12 del controlado C1 dejó de responder al ping continuo que estamos ejecutado. Esto significa que el controlador ya se encuentra fuera de servicio.

Figura 64

Prueba de conectividad hacia los 3 nodos del cluster, el segundo nodo ya no responde



Luego de la falla del 2do controlador, podemos ver en la figura 65 que los conmutadores se han divido entre los 2 controlador que quedaron operativos C0 y C2.

Figura 65

Lista de los nodos ONOS – 2do nodo fuera de línea

En la figura 66 se puede observar que el controlador C1 aparece como inactivo en el inventario de nodos del cluster ONOS.

Figura 66

Lista de los nodos de cluster ONOS – 2do nodo fuera de línea

RESULTADO DE LAS PRUEBAS

Finalmente, el resultado de las pruebas realizadas en este punto es satisfactorio

Tabla 16

Resultado de las pruebas de alta disponibilidad y balanceo de carga

PRUEBA	RESULTADO	COMENTARIOS
Formación del cluster	✓	El cluster se formó de manera satisfactoria,
Balanceo de carga	✓	Se verificó que los conmutadores se distribuyen de manera correcta entre los 3 nodos y ante la falla de uno, los conmutadores se distribuyen entre los 2 nodos que quedan operativos.
Continuidad del servicio ante la falla de un nodo	✓	De acuerdo con las pruebas realizadas, se verificó que el servicio continuó operando de manera correcta ante la falla de uno de los nodos.

5.4. Disminuir los tiempos en el despliegue de configuraciones

En este punto se demostrará el objetivo específico número 3 (Disminuir los tiempos de despliegue de configuraciones y actualizaciones de la red) y para lograrlo se realizará una comparación de los tiempos que demanda configurar los conmutadores en una red clásica o tradicional y una red bajo la arquitectura SDN y para ello se está considerando las situaciones siguientes:

PRUEBAS

Cómo parte de las validaciones, en busca de demostrar la reducción de los tiempos en el despliegue de las configuraciones en los conmutadores, se realizará las siguientes pruebas:

- ✓ Configuración de una nueva VLAN
- ✓ Cambiar la configuración de VLAN al puerto de un conmutador

5.4.1. Configuración de una nueva VLAN

Para este fin, se creará una VLAN haciendo uso del escenario de simulación en donde se verá la complejidad de configuración con SDN y los pasos que se emplearán para llegar a completar lo deseado.

Cómo podemos observar en la siguiente figura, antes de las configuraciones que se detallarán a continuación, ninguno de los 8 hosts tiene asignado una VLAN.

Figura 67

Inventario de hosts sin etiqueta de vlan

FRIENDLY NAME	HOST ID	MAC ADDRESS	VLAN ID	CONFIGURED	IP ADDRESSES
10.0.0.1	02:93:D1:E0:CD:41/None	02:93:D1:E0:CD:41	None	false	10.0.0.1
10.0.0.8	1E:6A:17:EE:BE:78/None	1E:6A:17:EE:BE:78	None	false	10.0.0.8
10.0.0.7	36:7D:08:6F:A1:C4/None	36:7D:08:6F:A1:C4	None	false	10.0.0.7
10.0.0.5	96:9A:95:39:08:E6/None	96:9A:95:39:08:E6	None	false	10.0.0.5
10.0.0.4	AE:8D:19:1A:2B:36/None	AE:8D:19:1A:2B:36	None	false	10.0.0.4
10.0.0.2	CE:3E:62:71:A1:CB/None	CE:3E:62:71:A1:CB	None	false	10.0.0.2
10.0.0.3	E2:2E:41:BA:F9:98/None	E2:2E:41:BA:F9:98	None	false	10.0.0.3
10.0.0.6	F6:09:EA:DA:68:FF/None	F6:09:EA:DA:68:FF	None	false	10.0.0.6

Cómo primer paso, se procederá a crear cada una de las VLAN en el entorno mininet, para lo cual ejecutaremos los siguientes comandos

```
mininet> h1 vconfig add h1-eth0 30
mininet> h2 vconfig add h2-eth0 40
mininet> h3 vconfig add h3-eth0 40
mininet> h4 vconfig add h4-eth0 30
mininet> h5 vconfig add h5-eth0 50
mininet> h6 vconfig add h6-eth0 80
mininet> h7 vconfig add h7-eth0 100
mininet> h8 vconfig add h8-eth0 60
```

El segundo paso consiste en eliminar las rutas que tenían los hosts de manera predeterminada, y se hace con los siguientes comandos:

```
mininet> h1 route del -net 10.0.0.0 netmask 255.0.0.0
mininet> h2 route del -net 10.0.0.0 netmask 255.0.0.0
mininet> h3 route del -net 10.0.0.0 netmask 255.0.0.0
mininet> h4 route del -net 10.0.0.0 netmask 255.0.0.0
mininet> h5 route del -net 10.0.0.0 netmask 255.0.0.0
mininet> h6 route del -net 10.0.0.0 netmask 255.0.0.0
mininet> h7 route del -net 10.0.0.0 netmask 255.0.0.0
mininet> h8 route del -net 10.0.0.0 netmask 255.0.0.0
```

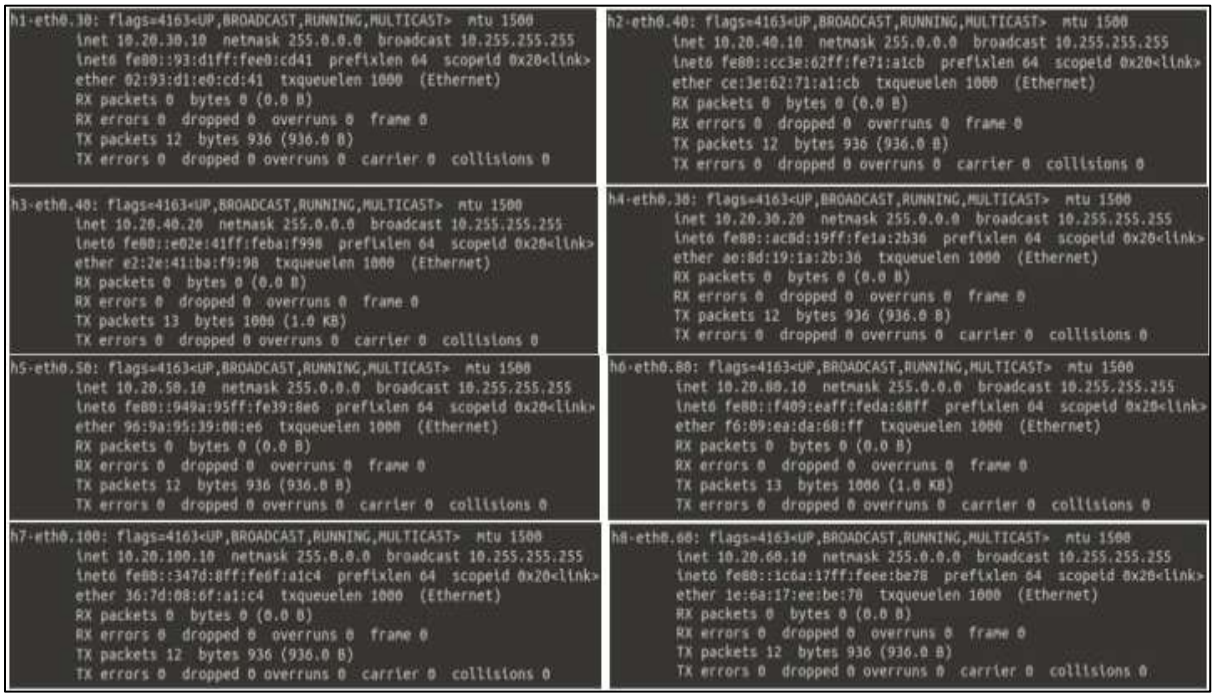
Tercero, agregamos el direccionamiento IP que le corresponde a cada uno de los hosts en la nueva VLAN. Para ello ejecutamos los siguientes comandos:

```
mininet> h1 ifconfig h1-eth0.30 10.20.30.10
mininet> h2 ifconfig h2-eth0.40 10.20.40.10
mininet> h3 ifconfig h3-eth0.40 10.20.40.20
mininet> h4 ifconfig h4-eth0.30 10.20.30.20
mininet> h5 ifconfig h5-eth0.50 10.20.50.10
mininet> h6 ifconfig h6-eth0.80 10.20.80.10
mininet> h7 ifconfig h7-eth0.100 10.20.100.10
mininet> h8 ifconfig h8-eth0.60 10.20.60.10
```

Con los 3 pasos anteriores se tiene configurado las VLAN en mininet; y como podrán ver, primero, la complejidad es prácticamente nula, es muy sencillo de ejecutar los comandos. Y segundo, el tiempo que nos ha demandado no es más de 1 minuto.

Figura 68

Configuración de los hosts con nuevo direccionamiento IP de sus respectivas vlan



En el cuarto paso se asignará la VLAN a cada una de las interfaces de los conmutadores en las cuales se encuentran conectados los hosts.

```
onos> interface-add -v 30 of:0000000000000006/2 h1
onos> interface-add -v 40 of:0000000000000006/3 h2
onos> interface-add -v 40 of:0000000000000007/2 h3
onos> interface-add -v 30 of:0000000000000008/2 h4
```

```
onos> interface-add -v 50 of:0000000000000008/3 h5
onos> interface-add -v 80 of:0000000000000009/2 h6
onos> interface-add -v 100 of:000000000000000a/2 h7
onos> interface-add -v 60 of:000000000000000a/3 h8
```

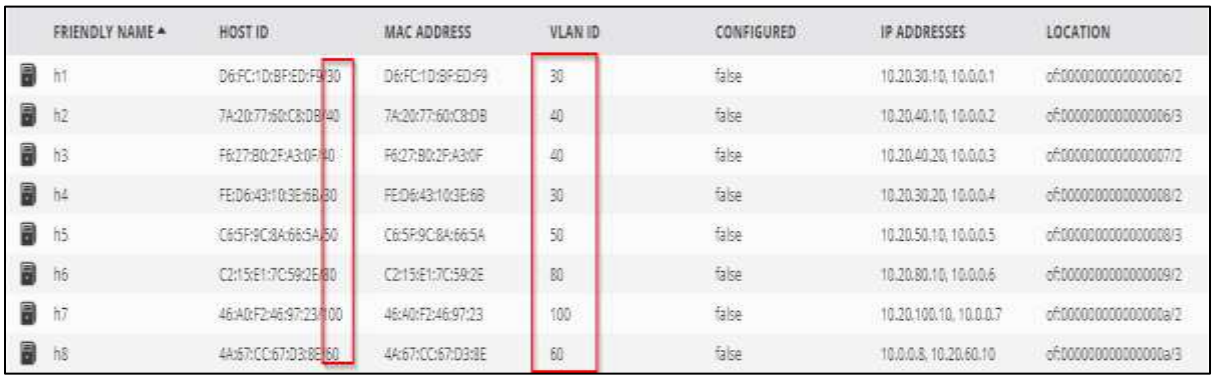
Finalmente, de acuerdo con el Capítulo 4, creamos un VPLS (Virtual Private Lan Service) y unimos todos los hosts al VPLS creado con los siguientes comandos:

```
onos> vpls create VPLS1
onos> vpls add-if VPLS1 h1
onos> vpls add-if VPLS1 h2
onos> vpls add-if VPLS1 h3
onos> vpls add-if VPLS1 h4
onos> vpls add-if VPLS1 h5
onos> vpls add-if VPLS1 h6
onos> vpls add-if VPLS1 h7
onos> vpls add-if VPLS1 h8
```

En la figura 69, se podrá observar que cada uno de los hosts aparece inventariado con su respectiva VLAN.

Figura 69

Inventario de hosts con etiqueta de vlan



FRIENDLY NAME	HOST ID	MAC ADDRESS	VLAN ID	CONFIGURED	IP ADDRESSES	LOCATION
h1	D6:FC:1D:BF:ED:F9/30	D6:FC:1D:BF:ED:F9	30	false	10.20.30.10, 10.0.0.1	of:0000000000000006/2
h2	7A:20:77:60:C8:DB/40	7A:20:77:60:C8:DB	40	false	10.20.40.10, 10.0.0.2	of:0000000000000006/3
h3	F6:27:B0:2F:A3:0F/40	F6:27:B0:2F:A3:0F	40	false	10.20.40.20, 10.0.0.3	of:0000000000000007/2
h4	FE:D6:43:10:3E:6B/30	FE:D6:43:10:3E:6B	30	false	10.20.30.20, 10.0.0.4	of:0000000000000008/2
h5	C6:5F:9C:8A:66:5A/50	C6:5F:9C:8A:66:5A	50	false	10.20.50.10, 10.0.0.5	of:0000000000000008/3
h6	C2:15:E1:7C:59:2E/80	C2:15:E1:7C:59:2E	80	false	10.20.80.10, 10.0.0.6	of:0000000000000009/2
h7	46:A0:F2:46:97:23/100	46:A0:F2:46:97:23	100	false	10.20.100.10, 10.0.0.7	of:000000000000000a/2
h8	4A:67:CC:67:D3:8E/60	4A:67:CC:67:D3:8E	60	false	10.0.0.8, 10.20.60.10	of:000000000000000a/3

Cómo se ha podido observar, el proceso para crear una nueva VLAN es completamente sencillo y el tiempo que demanda realizar la configuración es mínimo. Realizar esta configuración nos tomó menos de 10 (diez) minutos.

Lo siguiente que realizaremos, es la comparación del tiempo que toma ejecutar la configuración de una nueva vlan en una red tradicional frente a la red SDN de este proyecto. Los tiempos que mostraremos para la configuración en la red tradicional han sido tomados del capítulo 3, análisis del problema.

Tabla 17

Comparativo de tiempo en la configuración de una nueva vlan

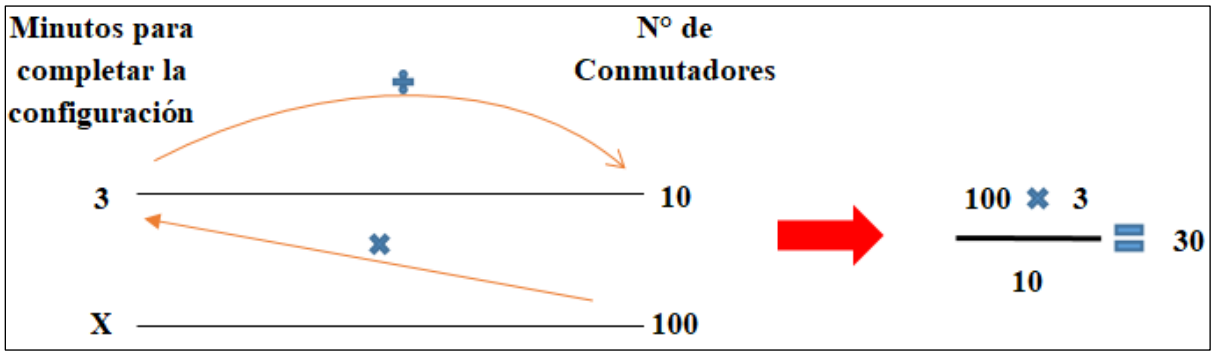
Actividad	Red Tradicional		Red SDN	
	Procedimientos	**Tiempo (min.)**	**Procedimiento**	**Tiempo (min.)**
Crear una nueva VLAN	Conectarse al conmutador Core y crear la VLAN	**30**	Crear la VLAN en mininet	**30**
	Identificar el(los) conmutador(es) y el(los) puerto(s) para crear y configurar VLAN Conectarse al(los) conmutador(es) para configurarlos	**60**	Identificar la interfaz del host y conmutador al que se debe configurar la nueva VLAN	**3**
	Configurar la VLAN en el(los) puerto(s) del conmutador de manera correcta	**30**	Configurar la interfaz y asignar los hosts a su respectivo VPLS	**7**
	Tiempo Total (min.)	**120**	**Tiempo Total (min.)**	**40**

Cómo se puede observar en la tabla 17, el tiempo que toma realizar la configuración de una nueva VLAN en la red SDN, frente al tiempo que se toma en una red tradicional es bajo, 40 minutos frente a 120 minutos respectivamente. El tiempo para completar la configuración de una nueva VLAN en la red SDN de este proyecto, escenario de pruebas, representa el **33.33%** respecto de una red tradicional.

En un escenario real de 100 conmutadores en la red SDN, lo único que cambiaría y podría llevar más tiempo en función de la cantidad de conmutadores, es el procedimiento de ***identificar la interfaz del host y conmutador al que se debe configurar la nueva VLAN***. En este sentido, si con 10 conmutadores, realizar la configuración, nos tomó 3 minutos; aplicando una regla de tres simple directa, tal como se muestra en la figura 70, con 100 conmutadores nos tomaría 30 minutos.

Figura 70

Regla de tres simple directa para calcular el tiempo de configuración de nueva VLAN con 100 conmutadores



Entones, de acuerdo con la tabla 16, considerando 100 conmutadores; el tiempo total que tomaría realizar la configuración sería de 67 minutos. Este tiempo representa el **55,83%** del tiempo que tomaría hacer la misma configuración en una red tradicional (120 minutos).

Inclusive, el tiempo expresado para realizar la configuración en el controlador, es posible realizarlo mediante la ejecución de un script. De esta manera, toda la configuración en el controlador se puede reducir a la ejecución de un único comando.

Primero crearemos un script de nombre **script_onos_add_int.sh**

```
# sudo vi script_onos_add_int.sh
```

Luego, editamos el contenido del srcipt. En este contenido veremos la conexión a onos mediante ssh por el puerto 8101 con las credenciales por defecto (usuario:onos y password:rocks); además, le indicamos las sentencias a ejecutar dentro de onos, en nuestro caso será la creación de 8 interfaces.

```
#!/bin/bash

sshpass -p "rocks" ssh -o StrictHostKeyChecking=no -p 8101 -X onos@10.20.20.11 "
interface-add -v 30 of:0000000000000006/2 h1;
interface-add -v 40 of:0000000000000006/3 h2;
interface-add -v 40 of:0000000000000007/2 h3;
interface-add -v 30 of:0000000000000008/2 h4;
interface-add -v 50 of:0000000000000008/3 h5;
interface-add -v 80 of:0000000000000009/2 h6;
interface-add -v 100 of:000000000000000a/2 h7;
interface-add -v 60 of:000000000000000a/3 h8;
vpls create VPLS1;
```

```
vpls add-if VPLS1 h1;
vpls add-if VPLS1 h2;
vpls add-if VPLS1 h3;
vpls add-if VPLS1 h4;
vpls add-if VPLS1 h5;
vpls add-if VPLS1 h6;
vpls add-if VPLS1 h7;
vpls add-if VPLS1 h8"
```

Para poder ejecutar la conexión con SSH, es necesario realizar la instalación de sshpass, y lo hacemos mediante el siguiente comando.

```
# sudo apt-get install sshpass
```

Para poder ejecutar el script, le damos permiso de ejecución

```
# sudo chmod +x script_onos_add_int.sh
```

Con el sshpass ya instalado procederemos a ejecutar el script.

```
# ./script_onos_add_int.sh
```

El resultado de ejecutar el script es el que aparece en la figura 71, ahí se puede ver que todas las interfaces han sido agregadas.

Figura 71

Resultado de la ejecución del sript

```
bagner@server-ONOS1:~/scripts$ ./script_onos_add_int.sh
Password authentication
Interface added
Interface added
Interface added
Interface added
Interface added
Interface added
Interface added
Interface added
      onos> interfaces
      h1: port=of:0000000000000006/2 vlan=30
      h2: port=of:0000000000000006/3 vlan=40
      h3: port=of:0000000000000007/2 vlan=40
      h4: port=of:0000000000000008/2 vlan=30
      h5: port=of:0000000000000008/3 vlan=50
      h6: port=of:0000000000000009/2 vlan=80
      h7: port=of:000000000000000a/2 vlan=100
      h8: port=of:000000000000000a/3 vlan=60
```

Este script lo hemos ejecutado manualmente. Sin embargo, con el propósito de automatizar la ejecución de las configuraciones y no depender de un administrador de red para que ejecute la configuración manualmente, en los siguientes pasos mostramos cómo podemos agendar la ejecución del script en un determinado horario de manera automática.

Para realizar la automatización en la ejecución de nuestro script, utilizaremos el administrador regular de procesos de nuestro sistema operativo Ubuntu, Cron. Cron administra los procesos en segundo plano (demonio) que ejecuta procesos a intervalos regulares (por ejemplo, cada minuto, día, semana o mes) haciendo uso de la tabla de tareas crontab.

Entonces, iniciamos editando nuestro crontab con el siguiente comando.

```
# crontab -e
```

El comando ***crontab –e*** inicia el editor en donde configuraremos la hora a ejecutar de nuestro script. En nuestro caso colocaremos que el script se ejecuta a las 00:30 horas de siguiente día.

```
00 30 * * * /home/bagner/scripts/script_onos_add_int.sh
```

Figura 72

Resultado de la programación del crontab – comando #crontab -l

```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
30 00 * * * /home/bagner/scripts/script_onos_add_int.sh
```

En la figura 72, se puede ver el resultado de la ejecución del ***crontab –l***, en esta imagen visualiza la hora en la cual se ejecutará este script. Es importante tener en cuenta que esa configuración dependerá de la hora y el día en que se quiera la ejecución del script.

5.4.2. Cambiar la configuración de VLAN al puerto de un conmutador

En este punto lo que realizaremos es cambiar de VLAN al host h3, este host inicialmente tal como podemos ver en la imagen 69, se encuentra en la VLAN 40 y lo cambiaremos a la VLAN 50.

Primero realizaremos el cambio en mininet, ejecutando los siguientes comandos para el cambio de VLAN y dirección IP de la VLAN 50.

```
mininet> h3 ifconfig h3-eth0.40 down
mininet> h3 vconfig add h3-eth0 50
mininet> h3 route del -net 10.0.0.0 netmask 255.0.0.0
mininet> h3 ifconfig h3-eth0.50 10.20.50.20
```

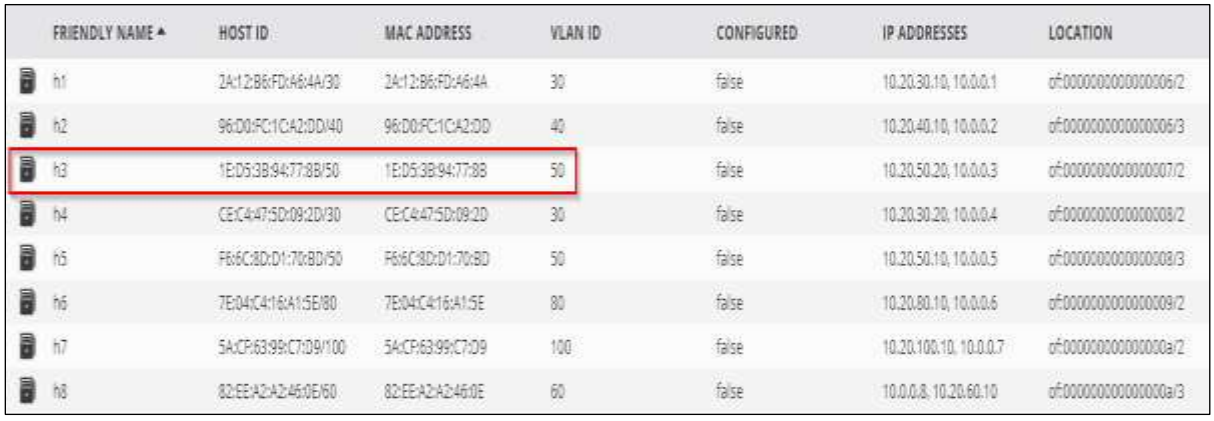
Luego en el controlador, se ejecutan los siguientes comandos, con los cuales, primero eliminamos la interfaz que se había creado en la VLAN 40, y luego volvemos a crear la interfaz, pero con la VLAN 50

```
onos> interface-remove of:0000000000000007/2 h3
onos> interface-add -v 50 of:0000000000000007/2 h3
```

En la figura 73, podemos ver que el host h3, ahora tiene VLAN ID 50

Figura 73

Reconfiguración de VLAN del Host h3



FRIENDLY NAME ▲	HOST ID	MAC ADDRESS	VLAN ID	CONFIGURED	IP ADDRESSES	LOCATION
h1	2A:12:B6:FD:A6:4A/30	2A:12:B6:FD:A6:4A	30	false	10.20.30.10, 10.0.0.1	of:0000000000000006/2
h2	96:D0:FC:1C:A2:DD/40	96:D0:FC:1C:A2:DD	40	false	10.20.40.10, 10.0.0.2	of:0000000000000006/3
h3	1E:D5:3B:94:77:8B/50	1E:D5:3B:94:77:8B	50	false	10.20.50.20, 10.0.0.3	of:0000000000000007/2
h4	CE:C4:47:5D:09:2D/30	CE:C4:47:5D:09:2D	30	false	10.20.30.20, 10.0.0.4	of:0000000000000008/2
h5	F6:6C:8D:D1:70:8D/50	F6:6C:8D:D1:70:8D	50	false	10.20.50.10, 10.0.0.5	of:0000000000000008/3
h6	7E:04:C4:16:A1:5E/80	7E:04:C4:16:A1:5E	80	false	10.20.80.10, 10.0.0.6	of:0000000000000009/2
h7	5A:CF:63:99:C7:D9/100	5A:CF:63:99:C7:D9	100	false	10.20.100.10, 10.0.0.7	of:0000000000000009/3
h8	82:EE:A2:A2:46:0E/60	82:EE:A2:A2:46:0E	60	false	10.0.0.8, 10.20.60.10	of:000000000000000a/3

El tiempo que nos ha tomado la ejecución de este procedimiento ha sido menos de 5 minutos.

Lo siguiente que realizaremos es la comparación del tiempo que toma realizar la configuración del cambio de vlan a un puerto en una red tradicional frente a la red SDN de este proyecto. Los tiempos que mostraremos para la configuración en la red tradicional han sido tomados del capítulo 3, análisis del problema.

Tabla 18

Comparativo de tiempo en la configuración de un puerto de conmutador para cambiar de vlan

Actividad	**Red Tradicional**		**Red SDN**	
	Procedimientos	**Tiempo (min.)**	**Procedimiento**	**Tiempo (min.)**
Reconfiguración de puertos en un conmutador (Cambio de VLAN)	Identificar el conmutador y el(los) puerto(s) para crear y configurar VLAN	**40**	Ejecutar el cambio en mininet	**40**
	Conectarse al(los) conmutador(es) para configurarlos Configurar la VLAN en el(los) puerto(s) del conmutador de manera correcta	**30**	Eliminar la interfaz con la vlan antigua y volver a crearla con la nueva vlan	**2**
	Tiempo Total (min.)	**70**	**Tiempo Total (min.)**	**42**

Cómo se puede observar en la tabla 18, el tiempo que toma realizar el cambio de vlan de un conmutador en la red SDN (42 minutos), frente a al tiempo que se toma en una red tradicional (70 minutos) es menor; este tiempo representa el **60%.**

RESULTADO DE LAS PRUEBAS

Finalmente, el resultado de las pruebas realizadas en este punto es satisfactorio

Tabla 19

Resultado de las pruebas Disminuir los tiempos en el despliegue de configuraciones

PRUEBA	RESULTADO	COMENTARIOS
Configuración de una nueva vlan	✓	El tiempo de configuración en una red SDN, respecto del tiempo de configuración en una red tradicional, representa el **55.83%.** También es importante recalcar que: primero, el tiempo descrito se puede reducir mucho más con la ejecución de la configuración mediante un script. Y segundo, la ejecución de los scripts se puede automatizar, de tal manera que no haya una ejecución manual.
Cambiar la configuración de vlan a un puerto de conmutador	✓	El tiempo de configuración en una red SDN, respecto del tiempo de configuración en una red tradicional, representa el **60%**

5.5. Propuesta de optimización de costos de inversión en la gestión centralizada de la red.

Una de las fortalezas de las redes SDN frente a las redes tradicionales es el costo que demanda para su gestión, tal como se ha mostrado en los capítulos anteriores, SDN plantea una mejora en cuanto a los costos de inversión en la implementación de un sistema de gestión de conmutadores centralizado.

PRUEBAS

Cómo parte de las validaciones, en busca de demostrar la fortaleza de la red SDN en cuanto a costos, se realizará la siguiente prueba:

- ✓ Comparación del costo para la gestión entre una red Cisco tradicional y una red SDN-ONOS

5.5.1. Comparación de los costos de gestión de una red tradicional (Cisco y Fortinet) y una red SDN-ONOS

Para hacer la comparación de los costos de gestión de una red Cisco tradicional y una red sobre SDN-ONOS elaboramos una tabla con todas las implicancias.

En relación con la figura 74, es importante tener en cuenta las siguientes consideraciones:

- ✓ En el caso de hardware, se está considerando el costo por arrendamiento de infraestructura tipo Leasing On-premise para la virtualización de los nodos en cada una de las soluciones. Este costo se paga de manera mensual y los costos expresados son por 1 año.

- ✓ El costo de soporte para el Cisco Prime Infrastructure es un pago de manera anual. Este costo se debe presupuestar anualmente, de tal forma que se asegure el soporte de parte del fabricante; sino se hace y no se cuenta con el soporte de la marca (Cisco), no se podrá tener el respaldo para solucionar problemas de fallas en la herramienta o mitigar vulnerabilidades de seguridad en la misma.

- ✓ El costo del sistema de gestión centralizada de Fortinet implica contar con el FortiManager (máquina virtual) y de una caja física (FortiGate) mediante los cuales se puede tener la gestión centralizada de todos los conmutadores de la red. Los costos están expresados como gasto mensual por los recursos de hardware para la máquina virtual, el soporte anual más la inversión inicial por la compra del equipo.

- ✓ En el caso de ONOS, está considerado el costo por 3 nodos, para la implementación de la alta disponibilidad. Mientras que en el caso de Cisco Prime Infratructure y Fortinet, solo se está considerando un nodo, es decir ante la falla de estos se perderá la gestión.

Figura 74

Comparativo de costos de inversión para implementar la gestión centralizada de una red

ITEM	Red Tradicional (CISCO)		Red Tradicional (FORTINET)		Red SDN	
	Sistema de gestión centralizado: (Cisco Prime Infrastructure)		Sistema de gestión centralizado: (FortiManager y FortiGate)		Sistema de gestión centralizado: (Controlador Open Networking Operating System)	
	Cantidad / Tipo	Costo	Cantidad / Tipo	Costo	Cantidad / Tipo	Costo
Licenciamiento	120	$12,600.00	-	-	Open Source	$0.00
Arrendamiento de Recursos de HW (Virtualización) x 1 año	CPU: 16 RAM: 24 GB HD: 1200 GB	$12,700.00	CPU: 16 RAM: 24 GB HD: 1200 GB	$12,700.00	CPU: 12 RAM: 12 GB HD: 300 GB	$9,500.00
Equipos	-	-	FortiGate + FortiManager	$20,481.82	-	-
Soporte	1 año	$12,409.72	1 año	$5,820.40	No aplica	$0.00
Nodos	1	-	1	-	3	-
COSTO TOTAL (CAPEX)	$12,600.00		$20,481.82		$0.00	
COSTO TOTAL (OPEX)	$25,109.72		$18,520.40		$9,500.00	
COSTO TOTAL EN 1 AÑO	**$37,709.72**		**$39,002.22**		**$9,500.00**	

Tal como se observa en la figura 74, el costo de inversión para implementar ONOS (**$9,500.00**) frente al costo de inversión para implementar el Cisco Prime Infrastructure (**$37,709.72**) representa el **28.43%.** Tomamos como referencia para el comparativo el costo de implementación con Cisco por ser menor que el de Fortinet,

RESULTADO DE LA PRUEBA

Finalmente, el resultado de la prueba realizada en este punto es satisfactorio

Tabla 20

Resultados de la prueba propuesta de optimización de costos de inversión en la gestión centralizada de la red

PRUEBA	RESULTADO	COMENTARIOS
Comparación del costo para la gestión entre una red Cisco tradicional y una red SDN-ONOS	✓	El costo por la implementación del controlador ONOS para la gestión centralizada respecto de la implementación de Cisco Prime Infrastructure para la gestión centralizada de una red tradicional Cisco representa el **28.43%**, lo cual representa un ahorro del **71.57%** de presupuesto.

6. CAPÍTULO 6: CONCLUSIONES Y RECOMENDACIONES

El presente proyecto contempla una red SDN para un campus universitario en el que no existe una gestión centralizada, principalmente, debido a los altos costos que significa invertir para implementar un sistema que gestione de manera centralizada a todos los conmutadores que conforman la red.

6.1. Conclusiones

1. En las dos pruebas realizadas sobre la creación de vlan y modificación de la vlan a un puerto de conmutador ha tomado 109 minutos, mientras que el tiempo utilizado para realizar las mismas configuraciones en una red tradicional es 190 minutos; es decir, se ha reducido 81 minutos respecto al tiempo que toma hacer estas configuraciones en una red tradicional. Esta reducción representa una optimización del 42.6% de tiempo; inclusive, este tiempo se puede optimizar aún más, con la ejecución de las configuraciones mediante la ejecución de scripts que pueden ser ejecutados manualmente o automatizados.

2. En la ejecución de las pruebas hemos podido comprobar que el controlador ONOS (Open Networking Operating System) mediante el cluster para la alta disponibilidad y su aplicación de balanceo de carga, funcionan de una manera muy óptima; a tal punto que ante la caída de un nodo del cluster, en menos de 3 segundos la cantidad de conmutadores de la red se redistribuyen entre los nodos de controlador que quedan disponibles; de esta manera, por un lado, los conmutadores no pierden gestión del controlador, y por otro lado, ninguno de los controladores se verá perjudicado por tener mayor número de conmutadores que gestionar.

3. En la prueba de comparación de costos para la gestión de la red, se evidenció que los costos de inversión para implementar la gestión centralizada de una red bajo la arquitectura SDN es de **$9,500.00** (dólares americanos), mientras que para tener el mismo nivel de gestión con el fabricante Cisco, el costo es de **$33,409.72** (dólares americanos); es decir, se tiene una reducción de **$23,909.72** (dólares americanos). Esta reducción representa una optimización del **71.57%** de presupuesto. De esta manera, los costos que se ahorran pueden ser designados a otros proyectos que ayuden en el monitoreo y/o mejoras de la red. Importante resultar que para esta comparación se tomó en cuanta los costos de Cisco por ser menor que los de Fortinet.

4. La investigación realizada y el análisis de la información nos permite concluir que la red SDN planteada en este proyecto, principalmente en cuanto a costos de inversión para la implementación de la gestión centralizada, muestra sus ventajas cuando se trata de una implementación nueva implementada desde cero; de lo contrario, podría parecer que la red SDN incrementa los costos para la gestión de la red.

6.2. Recomendaciones

1. En caso de implementar una red basada en la arquitectura SDN en un campus universitario, se recomienda utilizar el controlador ONOS, puesto que este controlador es muy sencillo de configurar y, además, al ser ONOS un controlador creado para proveedores de servicios de internet podría brindar múltiples bondades para universidades multicampus.

2. Para realizar la gestión de la red basada en SDN, es necesario que el administrador de red tenga conocimientos de desarrollo de software, principalmente, en los lenguajes de programación Python y Java que están estrechamente relacionados al controlador ONOS. Así mismo, también será necesario conocer de Linux, puesto que ONOS se instala en este tipo de sistemas operativos.

3. En las condiciones en las que se ha desarrollado este proyecto, se recomienda utilizar software libre para la implementación de la red SDN, pues representa una gran ventaja para crear nuevas aplicaciones e integrarlo al controlador y fortalecer la gestión de la red.

4. Cómo una forma de complementar el monitoreo de la red, se recomienda utilizar herramientas adicionales de código libre especializadas en el monitoreo de redes como, por ejemplo, CACTI, NAGIOS, etc.

7. REFERENCIAS

Alcívar, P., & Navia, M. (2020). Comparativa entre red tradicional y red definida por software: Caso de estudio ESPAM MFL. *RISTI - Revista Iberica de Sistemas e Tecnologias de Informacao*, *29*, 79–90. https://www.researchgate.net/publication/341622961_Comparativa_entre_red_tradicional_y_red_definida_por_software_Caso_de_estudio_ESPAM_MFL_Comparison_between_traditional_network_and_software_defined_network_Case_of_study_ESPAM_MFL

Ávila Martín, A. (2017, 10 de mayo). *Redes Definidas por Software (SDN), una tendencia en alza. - Sicrom*. Dan.com. Recuperado el 9 de diciembre de 2019, de https://sicrom.com/blog/redes-definidas-por-software-sdn/

Ayaka Koshibe. (2022, 21 de marzo). *Descarga de versiones de ONOS*. Docker Images. Recuperado el 5 de diciembre de 2022, de https://wiki.onosproject.org/display/ONOS/Downloads

Ccoyllo Sulca, I. (2018). *Redes definidas por Software (SDN)* [Diapositivas de PowerPoint]. Universidad Complutense Madrid. https://informatica.ucm.es/data/cont/media/www/pag-103596/transparencias/redes-por-software-SDN.pdf

Chafloque Mejía, J. D. (2018). *Propuesta de diseño de una red de datos de área local bajo la arquitectura de redes definidas por software para la Red Telemática de la Universidad Nacional Mayor de San Marcos* [Tesis de licenciatura, Universidad Mayor de San Marcos]. Repositorio institucional de tesis y trabajos de Titulación de la UNMSM. http://cybertesis.unmsm.edu.pe/handle/cybertesis/10017

Ching-Hao, C., & Ying-Dar, L. (2015). *OpenFlow Version Roadmap* [Trabajo de investigación, Espeed Network Lab]. Repositorio Espeed Network Lab. http://speed.cis.nctu.edu.tw/~ydlin/miscpub/indep_frank.pdf

Cision US Inc. (2013, 29 de agosto). *New Study Reveals: SDN Could Save Operators $4 Billion in Capital Expense by 2017*. About PR Newswire. Recuperado el 9 de diciembre de 2019, de https://www.prnewswire.com/news-releases/new-study-reveals-sdn-could-save-operators-4-billion-in-capital-expense-by-2017-221599341.html

España Tarapuez, N. E. (2016). *Diseño y simulación de una red definida por software (SDN)* [Tesis de licenciatura, Universidad Central del Ecuador]. Repositorio digital Academia. https://www.academia.edu/82308978/Dise%C3%B1o_y_simulaci%C3%B3n_de_una_red_definida_por_software_SDN_

Fernandez, R. (2020, 24 de noviembre). *Previsión: tráfico mundial en Internet por red de entrega de contenidos 2017-2022*. Statista. Recuperado el 28 de junio de 2023, de https://es.statista.com/estadisticas/635666/prevision-trafico-mundial-en-internet-por-red-de-entrega-de-contenidos/

García, A. C., Rodríguez, C. M. V., Anías, C. C., & Casmartiño, F. C. B. (2014). Controladores SDN, elementos para su selección y evaluación. *Revista Telem@tica*, *13*, 10–20. http://revistatelematica.cujae.edu.cu/index.php/tele

Gerometta, O. (2013, 9 de junio). *Los planos de operación de un dispositivo de red*. Blog Mis Libros de Networking. Recuperado el 17 de mayo de 2020, de http://librosnetworking.blogspot.com/2013/06/los-planos-de-operacion-de-un.html

Julio, I. (2015, 2 de setiembre). *Importancia de las redes de computadoras*. Blog Grupo 4 herramientas informática. Recuperado el 16 de enero de 2020, de https://grupo4herramientasinformatica.blogspot.com/2012/08/cipa-4-bloc-para-herramientas-de.html

Luo, L., & Wood, E. F. (2007). Monitoring and predicting the 2007 U.S. drought. *Geophysical Research Letters*, *34*(22). https://doi.org/10.1029/2007GL031673

Martínez Martínez, E., & El Vigía, E. (2016, 6 de octubre). *Arquitecturas de red LAN: breve historia*. El Vigía. Recuperado el 30 de enero de 2020, de https://www.elvigia.net/c-t/2016/10/6/arquitecturas-lan-breve-historia-250588.html

Mininet Project Contributors. (2018). *Mininet*. An Instant Virtual Network on your Laptop (or other PC). Mininet. Recuperado el 3 de abril de 2023, de http://mininet.org/

Newman, P., Edwards, W., Hinden, R., Hoffman, E., Liaw, F. C., Lyon, T., & Minshall, G. (1996). *Ipsilon's General Switch Management Protocol Specification Version 1.1*. Network Working Group. https://tools.ietf.org/html/rfc1987

Newman, P., Edwards, W., Hinden, R., Hoffman, E., Liaw, F. C., Lyon, T., & Minshall, G. (1998). *Ipsilon's General Switch Management Protocol Specification Version 2.0*. Network Working Group. https://tools.ietf.org/html/rfc2297

Onosproject. (2020, 15 de marzo). *ONOS Java API (1.13.10)*. Onosproject. Recuperado el 5 de diciembre de 2020, de http://api.onosproject.org/1.13.10/

Open Networking Foundation. (2013). *OpenFlow Switch Specification Version 1.4.0 (Wire Protocol 0x05)*. Open Networking Foundation. https://opennetworking.org/wp-content/uploads/2014/10/openflow-spec-v1.4.0.pdf

Open Networking Foundation. (2014a). *OpenFlow Switch Specification Version 1.5.0*. Open Networking Foundation. https://opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.0.pdf

Open Networking Foundation. (2014b). *SDN Migration Considerations and Use Cases ONF Solution Brief*. Open Networking Foundation. https://opennetworking.org/wp-content/uploads/2014/10/sb-sdn-migration-use-cases.pdf

Open Networking Foundation. (2015). *OpenFlow Switch Specification Version 1.5.1 (Protocol version 0x06)*. Open Networking Foundation. https://opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf

Pereira, G., & Gamess, E. (2017). Lineamientos para el Despliegue de Redes SDN/OpenFlow. *Revista Venezolana de Computación*, *4*(2), 21–33. https://www.researchgate.net/publication/333902840_Lineamientos_para_el_Despliegue_de_Redes_SDNOpenFlow

PowerData. (2016, 15 de setiembre). *La importancia de una buena gestión de redes de datos*. Blog PowerData. Recuperado el 16 de enero de 2020, de https://blog.powerdata.es/el-valor-de-la-gestion-de-datos/la-importancia-de-una-buena-gestion-de-redes-de-datos

Serrano Carrera, D. A. (2015). *Redes Definidas por Software (SDN): OpenFlow* [Tesis de licenciatura, Universidad Politécnica de Valencia]. Repositorio institucional UPV. https://riunet.upv.es/bitstream/handle/10251/62801/SERRANO%20-%20Redes%20Definidas%20por%20Software%20(SDN):%20OpenFlow.pdf?sequence=3

Sicrom. (2018). *Las características especiales de las SDN*. Dan.com. Recuperado el 15 de enero de 2020, de https://sicrom.com/blog/las-caracteristicas-especiales-de-las-sdn/

Superintendencia Nacional de Educación Superior Universitaria. (2023, 5 de abril). *La Superintendencia Nacional de Educación Superior Universitaria (SUNEDU)*. Sunedu. Recuperado el 8 de diciembre de 2019, de https://www.sunedu.gob.pe

Superintendencia Nacional de Educación Superior Universitaria. (2018). *Informe Bienal sobre la Realidad Universitaria Peruana 2018*. Sunedu. https://cdn.www.gob.pe/uploads/document/file/747830/Informe-Bienal-sobre-realidad.pdf?v=1590699157

Vega Guallpa, A. J., Andrade Cárdenas, D. P., & Pinos Castillo, L. F. (2022). Análisis comparativo de infraestructuras de redes SDN (Software Defined Networking) y redes tradicionales IP. *Pro Sciences - Revista de Producción Ciencias e Investigación*, *6*(43), 71–82. https://journalprosciences.com/index.php/ps/article/view/522

Walton, A. (2018). *Diseño Jerárquico de Redes*. Ccnadesdecero. Recuperado el 8 de diciembre de 2019, de https://ccnadesdecero.es/diseno-jerarquico-de-redes/

Yang, L., Corp., I., Dantu, R., Texas, Univ. of N., Anderson, T., Gopal, R., & Nokia. (2004). *Forwarding and Control Element Separation (ForCES) Framework*. Network Working Group. https://tools.ietf.org/html/rfc3746

8. GLOSARIO

Backup: Está referido a mantener un equipo de respaldo completamente operativo, en espera, ante la falla total o parcial de uno que está en operación.

Capa de Core: Es el nivel central y más alto de la red; su función principal es la de proporcionar transporte rápido y eficiente del tráfico que llega de los conmutadores de distribución para la comunicación entre diferentes redes o segmentos.

Capa de Distribución: Es la encargada de proporcionar comunicación entre las capas de acceso y Core. Se encarga de controlar el tráfico entre capas mediante políticas de acceso.(Walton, 2019)

Capa de Acceso: Es la capa más baja y se encarga de proporcionar acceso a la red para los grupos de trabajo y los usuarios y dispositivos finales en general. (Walton, 2019)

Controlador: Es una plataforma en la que se centraliza el plano de control de todos los dispositivos SDN.

Centro de Datos: Lugar diseñado bajo normas y estándares internaciones en donde se instalan equipamiento de redes de datos y otros equipamientos de Tecnologías de Información.

ISP: Es una empresa dedicada a proveer de servicios de internet a usuarios y empresas.

ONF: Open Networking Foundation (Fundación de redes abierta). Es un grupo que está enfocado en el desarrollo y estandarización de SDN. Además, propone una arquitectura dinámica, rentable y adaptable a la organización.

OpenFlow: Es un protocolo y elemento fundamental para la construcción de soluciones SDN.

QOS: Calidad de servicio. Es el rendimiento promedio que puede arrojar una red mediante una serie de pruebas y ello, le otorga prioridad al tráfico de datos de una organización.

Red LAN: Red de datos de área local que interconecta a dispositivos finales como computadoras, teléfonos, impresoras, etc. a través de uno o múltiples conmutadores en una misma ubicación física.

Red WAN: Red de computadoras de área amplia que interconecta a múltiples redes de área local sin la necesidad de que sus miembros no estén en la misma ubicación física. Estas redes se utilizan para interconectar dispositivos y recursos en diferentes ubicaciones o sitios, con lo cual se logra la comunicación y el intercambio de datos entre ellos.

SDN: Redes definidas por software. Es un nuevo paradigma en el ámbito de las redes que divide el plano de control y el plano de datos. Mediante la extracción del control de los conmutadores a un controlador externo para centralizarlo y puedan manejar la red de manera lógica o virtual.

SUNEDU: Organismo adscrito al Ministerio de Educación del Perú que cuenta con autonomía técnica, funcional, administrativa, económica y financiera. Entre sus principales funciones está la de licenciar a las universidades, filiales, facultades, escuelas y programas de estudios conducentes a grado académico.(SUNEDU, 2023)

Conmutador: Dispositivo de comunicación que permite la interconexión de equipos o terminales en la red de una organización (transporta datos mediante un procesamiento interno).

VLAN: Red de área local virtual. Permite crear redes lógicas totalmente independientes en una red física. Considerado un método que mejora la seguridad y gestión de los dispositivos de comunicación.

9. SIGLARIO

ISP: Internet Service Provider

LAN: Local Area Network

ONF: Open Network Fundation

SDN: Software Defined Network

SUNEDU: Superintendencia Nacional de Educación Superior Universitaria

VLAN: Virtual Local Area Network

WAN: Wide Area Network

AAA: Authentication, Authorization, and Accounting

LISP: Locator/ID Separation Protocol

LAC: Link Aggregation Control

NAT: Network Address Translation

ALTO: Application Layer Traffic Optimization

FaaS: Fabric as a service

NEMO: Network Mobility

NETCONF: Network Configuration Protocol

IoT: Internet of Things

SXP: SGT Exchange Protocol

10. ANEXOS

ANEXO A: Script de la red del escenario de simulación

```
#!/usr/bin/python

from mininet.net import Mininet
from mininet.node import Controller, RemoteController, OVSController
from mininet.node import CPULimitedHost, Host, Node
from mininet.node import OVSKernelSwitch, UserSwitch
from mininet.node import IVSSwitch
from mininet.cli import CLI
from mininet.log import setLogLevel, info
from mininet.link import TCLink, Intf
from subprocess import call

def myNetwork():

    net = Mininet( topo=None,
                   build=False,
                   ipBase='10.0.0.0/8')

    info( '*** Adding controller\n' )
    c0=net.addController(name='c0',
                         controller=RemoteController,
                         ip='10.20.20.11',
                         protocol='tcp',
                         port=6633)

    c1=net.addController(name='c1',
                         controller=RemoteController,
                         ip='10.20.20.12',
                         protocol='tcp',
                         port=6633)

    c2=net.addController(name='c2',
                         controller=RemoteController,
                         ip='10.20.20.13',
                         protocol='tcp',
                         port=6633)


    info( '*** Add switches\n')
    s1 = net.addSwitch('s1', cls=OVSKernelSwitch)
    s2 = net.addSwitch('s2', cls=OVSKernelSwitch)
    s3 = net.addSwitch('s3', cls=OVSKernelSwitch)
    s4 = net.addSwitch('s4', cls=OVSKernelSwitch)
    s5 = net.addSwitch('s5', cls=OVSKernelSwitch)
    s6 = net.addSwitch('s6', cls=OVSKernelSwitch)
    s7 = net.addSwitch('s7', cls=OVSKernelSwitch)
```

```
    s8 = net.addSwitch('s8', cls=OVSKernelSwitch)
    s9 = net.addSwitch('s9', cls=OVSKernelSwitch)
    s10 = net.addSwitch('s10', cls=OVSKernelSwitch)

    info( '*** Add hosts\n')
    h1 = net.addHost('h1', cls=Host, ip='10.0.0.1', defaultRoute=None)
    h2 = net.addHost('h2', cls=Host, ip='10.0.0.2', defaultRoute=None)
    h3 = net.addHost('h3', cls=Host, ip='10.0.0.3', defaultRoute=None)
    h4 = net.addHost('h4', cls=Host, ip='10.0.0.4', defaultRoute=None)
    h5 = net.addHost('h5', cls=Host, ip='10.0.0.5', defaultRoute=None)
    h6 = net.addHost('h6', cls=Host, ip='10.0.0.6', defaultRoute=None)
    h7 = net.addHost('h7', cls=Host, ip='10.0.0.7', defaultRoute=None)
    h8 = net.addHost('h8', cls=Host, ip='10.0.0.8', defaultRoute=None)

    info( '*** Add links\n')
    net.addLink(s1, s2)
    net.addLink(s3, s1)
    net.addLink(s3, s2)
    net.addLink(s4, s1)
    net.addLink(s4, s2)
    net.addLink(s5, s1)
    net.addLink(s5, s2)
    net.addLink(s6, s3)
    net.addLink(s7, s3)
    net.addLink(s8, s4)
    net.addLink(s9, s5)
    net.addLink(s10, s5)
    net.addLink(h1, s6)
    net.addLink(h2, s6)
    net.addLink(h3, s7)
    net.addLink(h4, s8)
    net.addLink(h5, s8)
    net.addLink(h6, s9)
    net.addLink(h7, s10)
    net.addLink(h8, s10)

    info( '*** Starting network\n')
    net.build()
    info( '*** Starting controllers\n')
    for controller in net.controllers:
        controller.start()

    info( '*** Starting switches\n')
    net.get('s1').start([c0,c1,c2])
    net.get('s2').start([c0,c1,c2])
    net.get('s3').start([c0,c1,c2])
    net.get('s4').start([c0,c1,c2])
    net.get('s5').start([c0,c1,c2])
    net.get('s6').start([c0,c1,c2])
    net.get('s7').start([c0,c1,c2])
```

```
    net.get('s8').start([c0,c1,c2])
    net.get('s9').start([c0,c1,c2])
    net.get('s10').start([c0,c1,c2])

    info( '*** Post configure switches and hosts\n')

    CLI(net)
    net.stop()

if __name__ == '__main__':
    setLogLevel( 'info' )
    myNetwork()
```